

Febrero 1986-300 ptas.
TodoSpectrum
AÑO 2.-NUMERO 18.
REVISTA EXCLUSIVA PARA USUARIOS
Descubrimos
el lenguaje C
Ordenadores en la escuela,
¿Para qué?
Curioseando en la ROM
QL MAGAZINE
Suplemento especial febrero 1986
Copy de grises
Microprocesadores 68000, una familia numerosa

ASI ES EL QL, HECHO PARA NOSOTROS



SENCILLO

Para los profesionales que necesitamos un teclado en nuestro idioma, QL nos ofrece, en castellano, su QWERTY standar de 65 teclas móviles.

Para los que deseamos comunicarnos a gran velocidad y capacidad con nuestro ordenador, QL nos presenta su lenguaje SUPER BASIC.



ASEQUIBLE

Para los que necesitamos gran margen operativo, ahora disponemos de un ordenador con memoria ROM de 32K que contiene el sistema operativo QDOS, un sistema mono-usuario, multi-tarea y con partición de tiempo.



PROFESIONAL

Para los que deseamos tener perfectamente ordenada nuestra agenda de trabajo, presupuestos, fichas de productos, nuestra correspondencia, estadísticas de venta, archivo... QL viene dotado de cuatro microdrives totalmente interactivados entre sí. QL QUILL de Tratamiento de Textos, QL ARCHIVE Base de Datos, QL ABACUS Hoja Electrónica de Cálculo y el QL EASEL para realización de todo tipo de gráficos.



ALGUNAS DE LAS CONFIGURACIONES MAS USUALES:

QL	QL MONITOR MONOCROMO	QL MONITOR MONOCROMO IMPRESORA	QL MONITOR COLOR	QL MONITOR COLOR IMPRESORA
PVP 79.500 PTS.	PVP 99.750 PTS.	PVP 149.750 PTS.	PVP 149.750 PTS.	PVP 199.750 PTS.



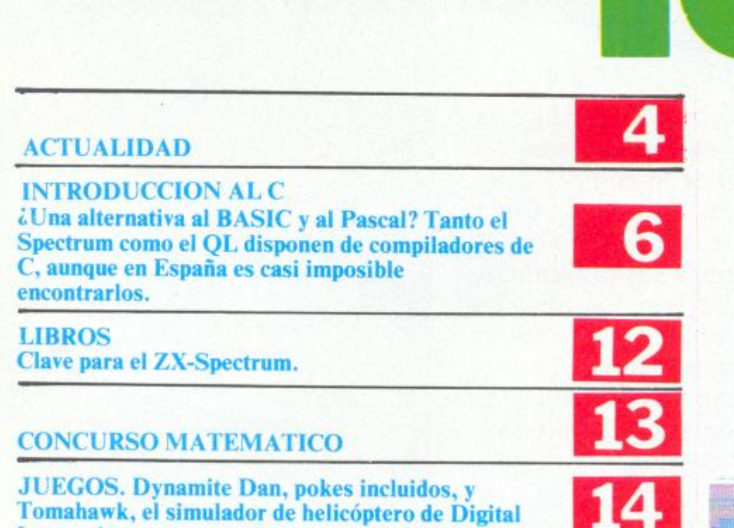
investronica

Tomás Bretón, 60. Telf. (91) 467 82 10. Telex 23399 IYCO E. 28045 Madrid
Camp. 80. Telf. (93) 211 26 58-211 27 54. 08022 Barcelona

etc

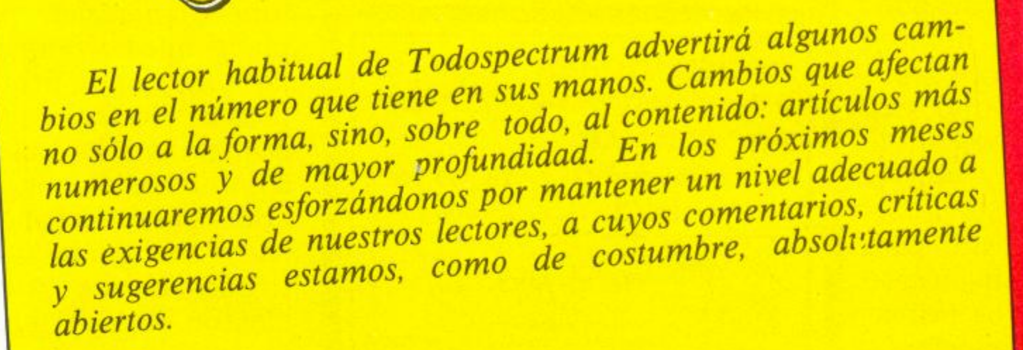
AÑO II

18



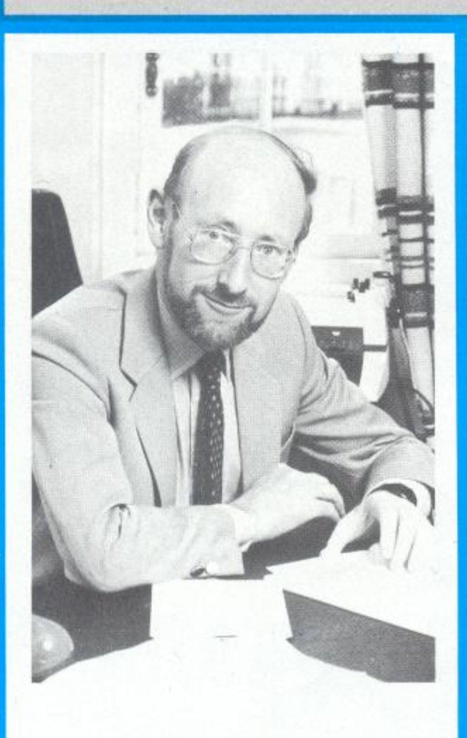






El lector habitual de Todospectrum advertirá algunos cambios en el número que tiene en sus manos. Cambios que afectan no sólo a la forma, sino, sobre todo, al contenido: artículos más numerosos y de mayor profundidad. En los próximos meses continuaremos esforzándonos por mantener un nivel adecuado a las exigencias de nuestros lectores, a cuyos comentarios, críticas y sugerencias estamos, como de costumbre, absolutamente abiertos.

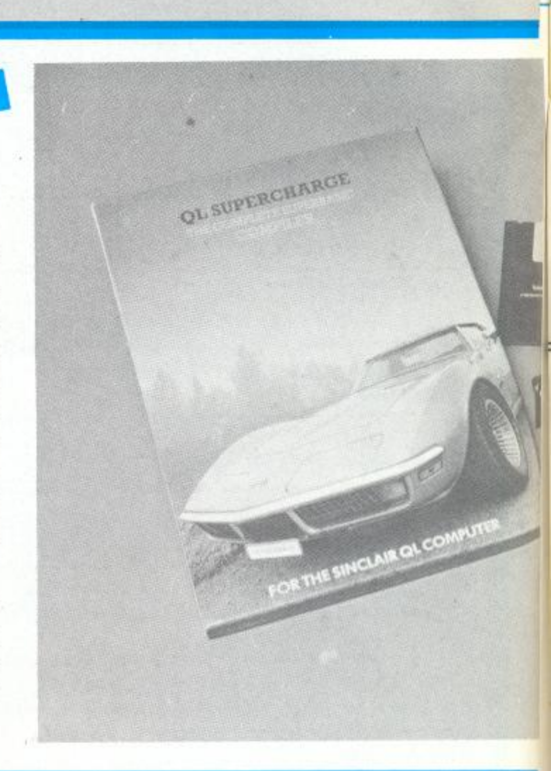
NOTICIAS



PROTECCION LENSLOK

El revolucionario sistema de protección basado en códigos de seguridad visibles mediante una lente es noticia constante. Cada vez son más las casas de software que lo incorporan a sus programas, aunque no todas con los mismos resultados. Firebird ha recibido numerosas llamadas de enfurecidos clientes incapaces de utilizar el Elite a causa de las pésimas instrucciones de Lenslok.

También los productores de software para QL están implantando este sistema, como Digital Precision en su compilador de BASIC Supercharge.



SINCLAIR DESMIENTE LA EXISTENCIA DE UN SPECTRUM CP/M

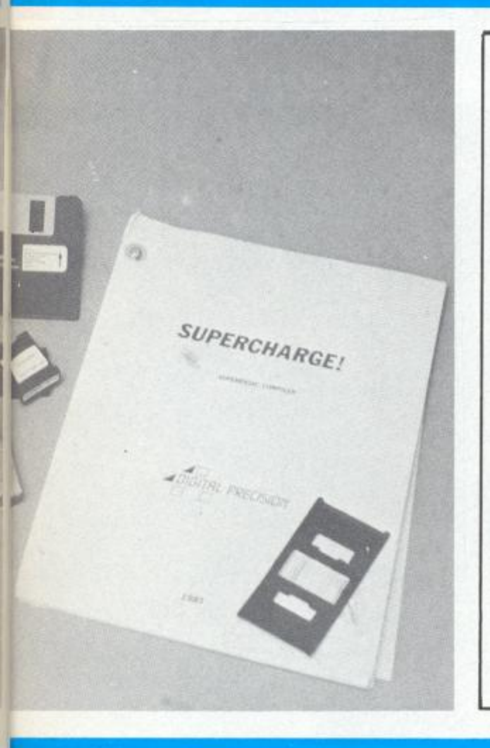
Sir Clive negó recientemente los rumores sobre el lanzamiento de una nueva versión de Spectrum capaz de trabajar en CP/M. Asimismo, aseguró que no dotará a sus máquinas con este sistema operativo hasta la aparición de su ordenador portátil. El Pandora, nombre con el que se le conoce desde hace ya bastante tiempo, será posiblemente compatible con el Spectrum y con el operativo CP/M. Su lanzamiento está previsto para este año, pero en ningún caso se producira antes de primavera.

LOS INGLESES SIGUEN ESPERANDO

Pese a que la aparición del Spectrum 128 en el mercado británico parecía inminente, Sinclair no ha desvelado todavía la fecha de su presentación. Mientras tanto, varias casas de *software* trabajan para la nueva máquina y en España comienzan a aparecer los primeros juegos, en su mayoría adaptaciones de los ya existentes, como el Super Test de Ocean.

FERRANTI, ALGO MAS QUE ULAS

El nombre Ferranti le resultará familiar a quien alguna vez haya observado el interior de su Spectrum. Efectivamente, la ULA lleva el sello de esta empresa británica. Sin embargo, muchos lectores ignorarán probablemente que Ferranti es una de las principales firmas europeas de electrónica y que recientemente ha obtenido un contrato de siete millones de libras para diseñar un ordenador de bajo consumo destinado a los submarinos Oberon de la Armada británica.



RENACE EL OPTIMISMO EN SINCLAIR RESEARCH

Tras la crisis financiera vivida durante el pasado año fiscal (pérdidas por valor de 18,3 millones de libras), Sinclair Research ha superado sus principales problemas y se dispone al lanzamiento de nuevos productos.

Entre los objetivos prioritarios de la empresa figura el conservar su actual cuota de mercado, que en el Reino Unido supera el 54 por 100 y continúa creciendo. El volumen de ventas alcanzó los 102,8 millones de libras frente a los 77,7 del año anterior. También crecen las exportaciones, que representan el 40 por 100 de las ventas de Sinclair.

La situación del pasado año fue más dura debido al fracaso de Sinclair Vehicles, lo que le costo a Sir Clive una buena parte de su fortuna personal. Por si fuera poco, su sueldo en Sinclair Research paso de 77.000 libras a «sólo» 53.000 libras, es decir unos 11,5 millones de pesetas.

¡WHAM! THE MUSIC BOX

Este es el título del último programa producido por Melbourne House, que permite componer música con el Spectrum. Para darse una idea de sus posibilidades basta escuchar el sonido de Fairlight, que utiliza la misma rutina. Trabaja con cuatro octavas y es capaz de manejar dos voces.



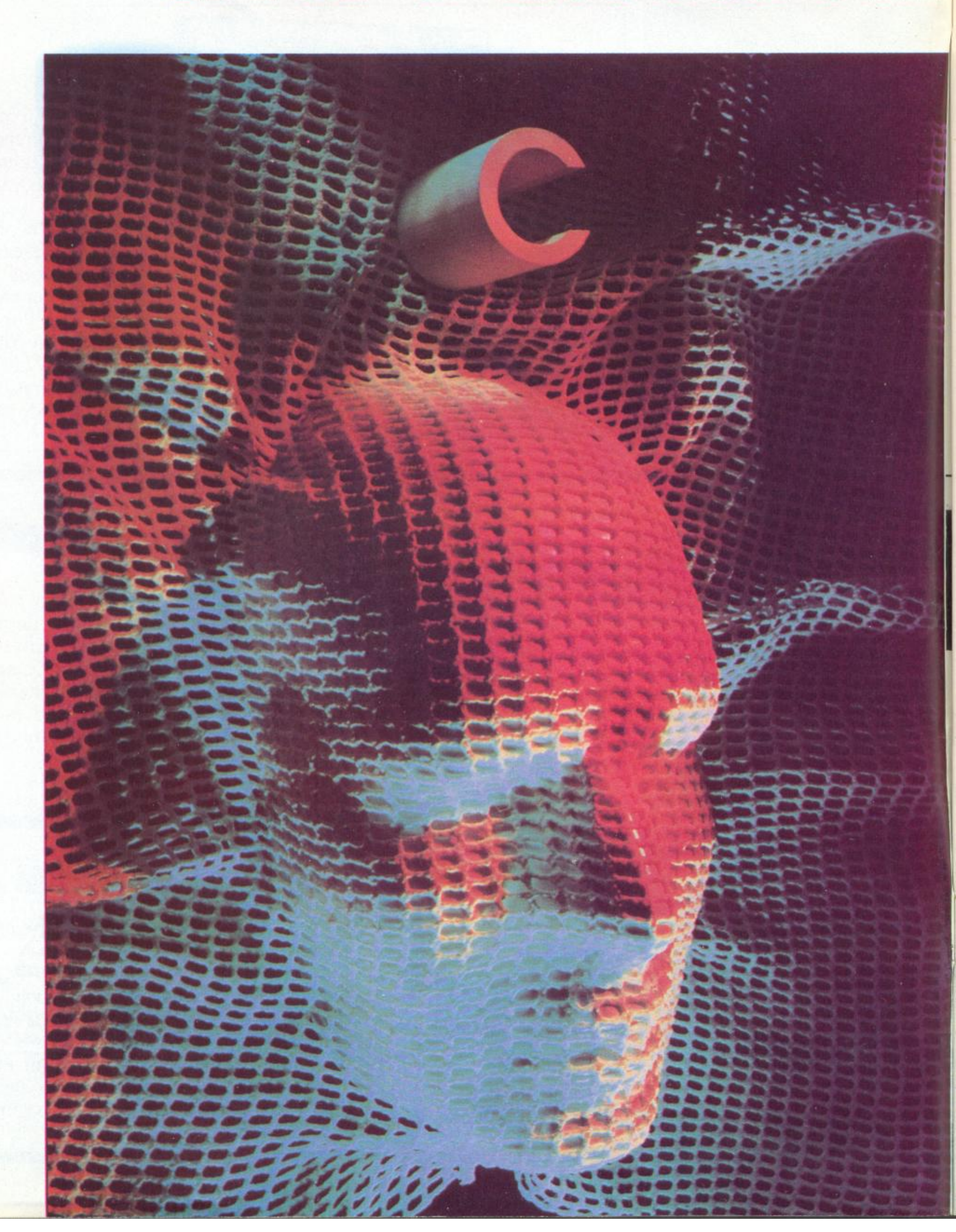
EL QL YA TIENE IMPRESORA

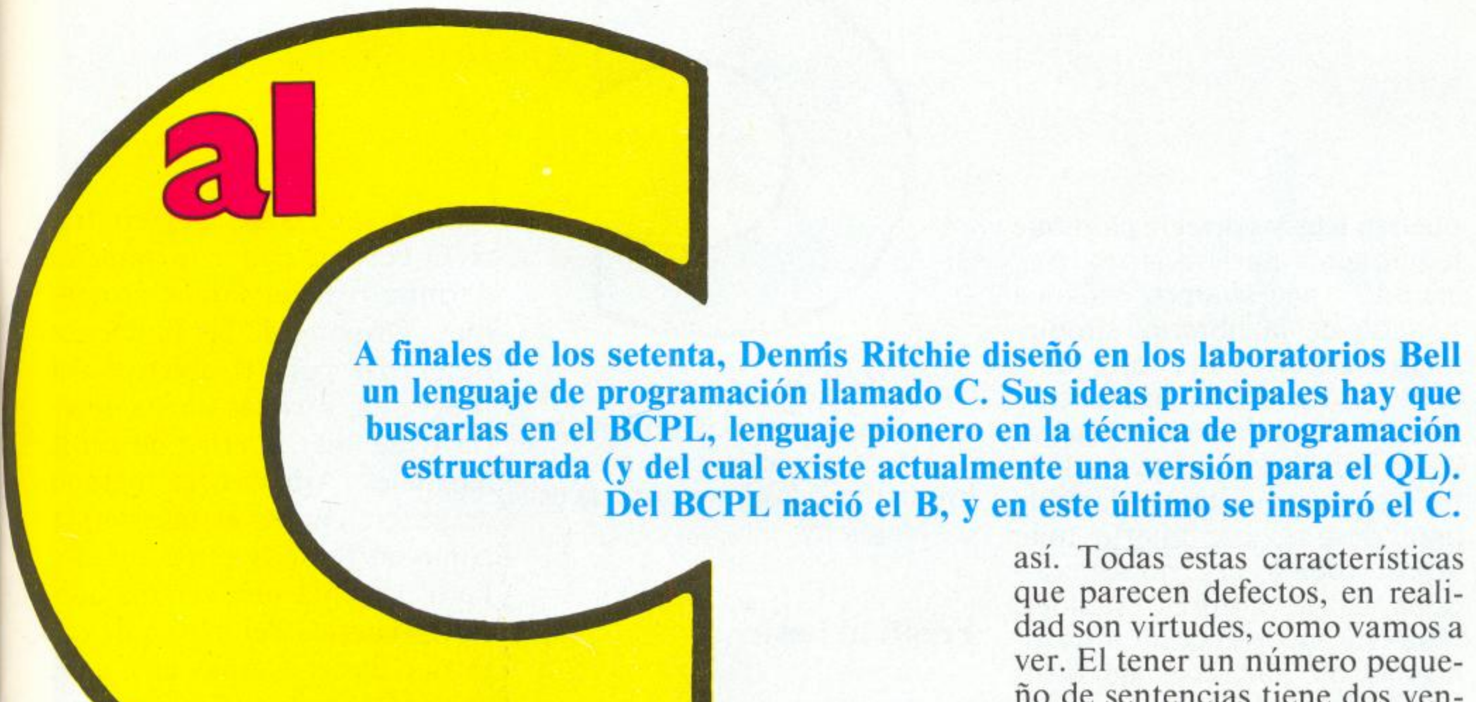
Sinclair Research ha seleccionado como impresora oficial del Ql una Seikosa de matriz de puntos, con una cabeza de impresión de nueve agujas. Su precio aproximado es de 250 libras y no necesita ningún interface para conectarse al QL. Su velocidad es de 100 caracteres por segundo en modo normal y 25 en alta calidad.

EPROMS PARA QL

Anglo Services ha desarrollado el primer programador de EPROMs para QL. Acepta una amplia gama de memorias EPROM, desde las 2516 y 2716 de 2 Kbytes, hasta las nuevas 27256 de 32 Kbytes. La placa se conecta al port de usuario del QL. El *software* suministrado puede transferirse fácilmente a disco, suponiendo que se disponga de un adaptador para mantener conectados simultáneamente ambos periféricos.

Introducción





A finales de los setenta, Dennis Ritchie diseñó en los laboratorios Bell un lenguaje de programación llamado C. Sus ideas principales hay que buscarlas en el BCPL, lenguaje pionero en la técnica de programación estructurada (y del cual existe actualmente una versión para el QL). Del BCPL nació el B, y en este último se inspiró el C.

C se diseñó para un sistema operativo llamado UNIX, uno de los más potentes y usados hoy en día. En realidad este sistema y todos su *software* estaba, como lo sigue estando en su versión más moderna, escrito en C. Debido a esta aplicación, se piensa del C que es un lenguaje muy especializado y orientado hacia la escritura de complicados sistemas operativos. Sin embargo, la realidad es otra: el C es un lenguaje de objetivo general, es decir, igualmente orientado hacia simples aplicaciones, como a complicados paquetes. Con C se realizan infinidad de programas, desde tratamiento de texto, a cálculos numéricos, cubriendo entre ambos un amplio campo de utilidades. De hecho, esta es la principal cualidad del C, su gran versatilidad.

Cómo es el C

C es un lenguaje compilado, es decir, se escribe el programa mediante un editor, y después un compilador y un montador nos dan un programa ejecutable, en código máquina. Tiene muchas similitudes con el Pascal, pero su estructuración no es tan rígida como la de éste. C es un lenguaje conciso, no tiene gran cantidad de sentencias, se podría decir que las justas. Puede manejar toma de decisiones (IF-ELSE), puede controlar lazos con condiciones (FOR, WHILE, DO) y puede ejecutar sentencias selectivamente (SWICHT), y poco más. Trabaja con diversos tipos de variables que han de ser declarados, igual que en Pascal (enteras, caracteres, doble precisión, etc.).

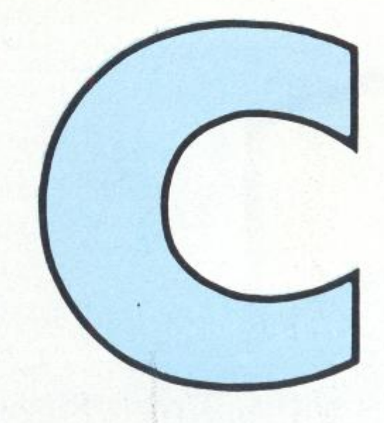
¿Y esto es todo?

Por lo visto hasta ahora, no parece que el C sea nada del otro mundo. Además, hay que decir que tampoco tiene sentencias propias de entrada y salida, ni de manejo de ficheros, ni ninguna función. Ahora parece estar peor que antes, pero no es así. Todas estas características que parecen defectos, en realidad son virtudes, como vamos a ver. El tener un número pequeño de sentencias tiene dos ventajas: primero, el lenguaje es breve, y por tanto muy fácil de aprender y manejar. Además, su compilador es muy reducido y por lo tanto ocupa poca memoria. Este aspecto resulta fundamental en los sistemas que deben mantener simultáneamente en memoria el programa escrito y el compilador, como sucede en todos los micrordenadores, por ejemplo el Spectrum.

Cómo es y cómo funciona el más versátil y potente de los últimos lenguajes de programación.

Las librerías del C

Y, ¿qúe pasa con las órdenes de E/S, manejo de ficheros y las funciones? No hay ningún problema, todos los C vienen acompañados de una librería estándar con todas aquellas funciones que podemos necesitar y que el C no tiene como propias. Además todas estas funciones están escritas en el propio lenguaje, por lo tanto se



pueden leer y corregir para que funcionen a nuestro gusto. Así, cuando necesitamos alguna función de la librería, simplemente se escribe en el programa como si fuese una instrucción más. El compilador se encargará de incluir todo el texto de la función en nuestro programa, para después compilarlo todo junto. Además el C no sólo dispone de la librería estándar, también puede acceder a las librerías de funciones de otros lenguajes.

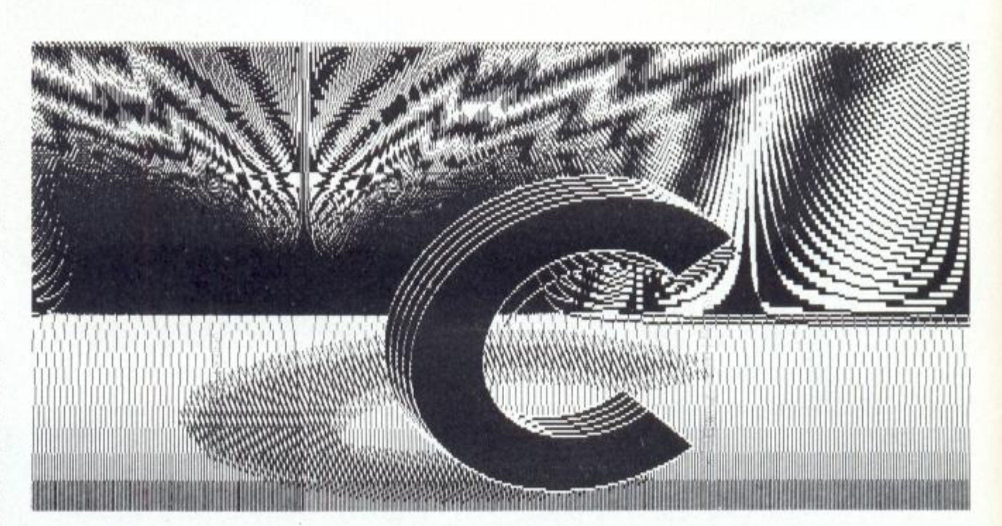
Comparando

Por lo visto hasta este punto, parece que hay un empate entre el C y los demás lenguajes conocidos. Hacen más o menos las mismas cosas, y funcionan más o menos igual. Para romper el empate vamos a compararlos con el C. El Cobol y el Fortran son dos lenguajes fuertemente especializados. Surgieron de problemas muy concretos (la gestión de datos comerciales y el trabajo con complicadas fórmulas numéricas), ambos tienen unas estructuras muy rígidas y una cantidad ingente de comandos, una para cada posible necesidad de cada campo de aplicación. Frente a ellos el C ofrece una estructura libre que permite al programador trabajar «a su aire». Es mucho más fácil de aprender, por su corto número de sentencias. Y ofrece por el acceso a su librería todas las funciones de estos lenguajes.

El sistema operativo UNIX está escrito en C. La programación en C es tan simple como en Basic. Un programa C funciona en todos los ordenadores que soporten C.

Frente al Basic

Los anteriores eran lenguajes mono-propósito, con un único objetivo. Veamos cómo es el lenguaje multi-propósito por excelencia, el Basic. Puede realizar cálculos matemáticos y puede manejar información alfanumérica. Puede controlar ficheros y puede dibujar gráficas, parece ser el lenguaje universal. Pero su capacidad de cálculo y manejo de información tienen un límite. Además, suele ser un lenguaje interpretado. El programa escrito reside en memoria y cuando se ejecuta se va traduciendo la sentencia en curso. Esto ocasiona un problema de memoria que limita la extensión de los programas que pueden realizarse. Por contra, el C con su pequeño tamaño permite realizar largos programas. Dispone de las funciones necesarias para el objetivo del programa, sacadas de su librería o de sus librerías de otros lenguajes. Además, el método de programación es tan simple como en Basic, y estructurado. Pero presenta una ventaja adicional surgida del mayor defecto del Basic: C sólo hay uno, con más o menos funciones en sus librerías, pero sólo uno. Un programa C funciona igual en todos los ordenadores que soporten C. Sin embargo, de Basic hay mil y una versiones diferentes, casi tantas como modelos de ordenadores hay. Así, hay que escribir nuevos programas, si nos decidimos a cambiar de ordenador.



Balance

Comparar C con el Pascal no resolverá ninguna duda. Realmente son dos lenguajes muy iguales, tienen el mismo aspecto y funcionan de forma pareci-

da. El Pascal parece un hermano mayor de C, más serio, más rígido, más estructural, de mayor nivel. El C como hermano menor es más libre, más corto, menos estructurado, más cerca de la máquina. Son similares, ambos hijos de una misma madre, la última o penúltima generación de lenguajes de programación, por eso sus cualidades son parecidas. Resumiendo, el C es un lenguaje conciso, con pocas sentencias que aprender y un compilador que ocupa poca memoria. No es nada especializado, lo cual le permite trabajar en cualquier campo, verdaderamente multipropósito. Con una estructura no rígida, fácil de manejar. Con unas posibilidades muy amplias gracias a sus librerías de funciones. Y es completamente estándar: un programa C funciona en cualquier ordenador que soporte el lenguaje.

Exito del C

Un lenguaje puede ser potente pero arduo de utilizar (P. ej.: ENSAMBLADOR). Puede ser específico, pero exigir una gran disciplina en la forma de escribir los programas (p. ej.: COBOL). Y sin embargo, estos dos lenguajes son ampliamente utilizados, a pesar de sus defectos. El éxito de un lenguaje no se mida en su potencia, ni en su idoneidad para un trabajo. Lenguajes existen muchos, pero aceptados universalmente, muy pocos. ¿Cuál es la clave de su éxito?: El programador. Si se siente cómodo con un lenguaje, ese lenguaje habrá triunfado. El C posee esta característica. El programador se siente muy cómodo trabajando en C. Puede

CUADRO 1

Veamos cómo sería, en Basic y en C, un programa que acepte por teclado una cadena de sólo 10 caracteres.

BASIC:

```
10 DIM A&(10)
20 FOR N=1 TO 10
30 IF INKEY$<>"" THEN GOTO 30
40 IF INKEY$="" THEN GOTO 40
50 A$(N)=INKEY$
60 NEXT N
```

C:

```
main()
(
        char a[10];
        int n,c;
        for(n = 0; n < 10; ++n)
        (
                c = getchar();
                a[n] = c;
        )
        a[10] = '/0'
)
```

Ahora veamos la creación de la función «parte entera» de forma artesana. El ejemplo en C, puede ser almacenando en una librería. Para utilizarlo en otros programas que quieran esa función.

BASIC:

```
10 INPUT A
20 IF A>0 THEN GOTO 60
30 FOR N=0 TO A-1 STEP -1
40 IF A<N and A<N-1 THEN ENT=N: GOTO 90
50 NEXT N
60 FOR N=0 TO A+1
70 IF A>N AND A<N+1 THEN ENT=N: GOTO 90
80 NEXT N
90 PRINT N
```

C:

```
ent(n)
flotat n;
c
        int c;
        c = 0;
        if(n < 0)
        (
                while (c < n && n < c+1)
                        —c;
)
else
(
      while(c < n && < c+1)
            +c;
)
return(c)
```



hacer de él lo que quiera, con total libertad. Desde usarlo como un Basic o un Pascal (con sus funciones), hasta hacer de él un semi-ensamblador.

Alto y bajo nivel

El C sólo exige definir los tipos de las variables a utilizar en nuestro programa. A partir de ahí todo es libre. Sólo tenemos que escribir el programa. Podemos ir llamando a las funciones que necesitemos y el compilador se encargará de montarlas en nuestro programa. Es un trabajo alejado de la máquina, en alto nivel. También podemos ir creando nuestras propias funciones: entradas y salidas, peticiones de datos, funciones de cadenas y funciones matemáticas. Así, creamos nuestra propia librería de funciones, que podemos ir ampliando y utilizar en otros programas. Es un trabajo cercano a la máquina, en bajo nivel.

El C ofrece una estructura libre que permite al programador trabajar «a su aire».
C es breve; pocas sentencias que aprender y manejar.

A nuestro aire

Se puede personalizar el lenguaje. Incluso crear un nuevo sublenguaje que nos guste más, todo ello a base de crear funciones, que serán los comandos del nuevo dialecto. Esta es la gran cualidad del C, su libertad. Podemos llegar a dominar la máquina, sin demasiadas complicaciones, y con pocas instrucciones que aprender. El C ofrece mucho y no pide demasiado. Si te gusta programar, te encantará.

Joaquín Mateos Lago

Bibliografía

— *El lenguaje de programación C.* Kernighan Ritchie. Prentice Hall, 1978.

— *Learning to program in C.* Thomas Plum. Prentice Hall, 1983.

CUADRO 2

Ahora un programa que calcula el factorial de un número. Se puede apreciar que el C y el Pascal tienen muchas similitudes. El programa en C puede utilizarse también como una función para otros programas.

Pascal:

```
Program factorial (input, output);
var factorial: integer;
   n           integer;
begin
     read(n);
     factorial :=1;
     while n > 0 do
         begin factorial := factorial * n;
              n          := n–1;
     end;
   write (/el factorial de/, n : 5, /es:/, factorial)
end
```

C:

```
factorial(n)
int n;
(
      int c;
      int factorial;
      factorial = 1;
      for(c = 2; c < n+1; ++c)
      (
              factorial=factorial*c;
      )
      printf("el factorial de %10d es: %10dd /n", n, facto-
      rial);
)
```





PIN SOFT, S.A.

Paseo de Gracia, 11 - Esc. C., 2.º 4.ª
Tel. (93) 318 24 53 - 08007 Barcelona

VIDEOJUEGOS (SPECTRUM)

ABU SIMBEL (PROFANATION)	1.875
AUTOMANIA	1.563
BASEBALL	1.607
BEACH HEAD	1.563
BOULDER DASH	1.563
BOUNTY BOB STRIKES BACK	1.875
BUCKROGERS	1.964
COMANDO	2.181
CYCLONE	1.563
CRITICAL MASS	1.696
DAM BUSTERS	1.964
DARTZ	1.563
DUMMY RUN	1.875
DEATHCHASE	1.563
DRAGONTORC	1.696
FIGHTING WARRIOR	1.875
FRANK BRUNO'S BOXING	1.991
FRANK N. STEIN	1.563
FULL THROTTLE	1.563
GHOSTBUSTERS	1.786
GRAND NATIONAL	1.991
GREMLINS (Castellano)	2.054
GYROSCOPE	1.696
HIGHWAY ENCOUNTER	1.696
HYPERSPORTS	1.875
JET SET WILLY	1.607
JACK AND THE BEANSTALK	1.563
JASPER	1.161
MAPGAME	2.455
MATCH POINT	1.563
MAZIACS	1.563
MOON ALERT	1.563
NIGHT GUNNER	1.563
OLE TORO	1.875
OLYMPICON	1.563
ORC ATTACK	1.563
POGO	1.563
POLE POSITION	1.607
PYJAMARAMA	1.563
RAID OVER MOSCOW	1.875
RAMBO	1.875
ROCKY	1.607
SABOTEUR	1.696
SGRIZAM	1.741
SHADOW OF THE UNICORN	3.830
SHADOWFIRE	1.429
SOUTHERN BELLE	1.875
SQUASH	1.991
STRONG MAN	2.085
SUPERMAN	2.464
SUPERTEST	1.875
TAPPER	1.875
THE WAY OF EXPLODING FIST	2.054
THEY SOLD A MILLION	2.232
TORNADO LOW LEVEL-TLL	1.563
TRAVEL WITH TRASHMAN	1.563
UGH !!	1.563
WANTED MONTY MOLE	1.161
WEST BANK	1.741
WORLD CUP	1.563
WORSE THINGS HAPPEN AT SEA	1.563
WORLD SERIES BASKETBALL	2.054

AMSTRAD

HARDWARE	
CABLE AMSTRAD-IMPRESORA	4.000
CABLE AMSTRAD 6128-CASSETTE	1.000
SOFTWARE	
COPY RITEMAN F+ Y SP-1000	3.500
AMSWORD II	6.500
MASTERCALC	6.500
VIDEOJUEGOS	
BASEBALL	1.875
BOULDER DASH	2.464
D.T. DECATHLON	1.875
EVERYONE'S A WALLY	2.200
FRANK BRUNO'S BOXING	1.991
GREMLINS	2.054
HERBERT'S DUMMY RUN	1.875
RAID	2.054
ROCKY	1.875
SOUTHERN BELLE	2.054
STRONG MAN	2.085
THEY SOLD A MILLION	2.232
3D VOICE CHESS (DISCO)	3.300

EN PREPARACION PARA AMSTRAD SITI CP/M *

*Servicio de Adaptación ficheros SITI Spectrum.

SOFTWARE SPECTRUM

S.I.T.I. V.3* 4.000

Al comprar esta versión abonamos 3.000,— ptas por cualquier versión anterior.

Context V.9* 4.000

Tratamiento de Textos.
Funciona con cualquier impresora.
Cassette y/o microdrive. 64 col. en pantalla e impresora.
Acentos graves y agudos.
Al comprar esta versión abonamos 3.000,— Ptas. por cualquier versión anterior.

Adaptador SITI-CONTEXT 2.500

Permite pasar información del SITI al CONTEXT.

M.D.S.-Sistema Operativo para Microdrive 7.000

Conjunto de nuevos comandos BASIC que permiten Acceso Aleatorio a Ficheros en Microdrive con un tiempo medio de acceso de 4 segundos.

CONTABILIDAD PIN* 3.000

Plan contable 200 cuentas, 2000 asientos. Hasta 9.000.000.000. Balance con activo-pasivo, cta. resultados. Utiliza el S.O.M.D.S. Cualquier impresora 80 col.

Kit Utilidades Discovery 1 3.000

10 utilidades CAT extendido. ON ERROR. Set de caracteres del Amstrad, etc.

AJUSTE DE CABEZALES CASSETTE 2.500

SINTETIZADOR DE VOZ	**3.000**
MULTI-COPYS (Copys desde 2 cm. hasta 70 cm.)	**3.000**
COPY GRISES (F+, SP-800, SP-1000, GP-550)	**2.500**
COPY RS-232	**2.500**
COPY SERIE RITEMAN F+	**2.500**
EDITOR 64 (64 columnas en pantalla)	**2.750**

*** Disponible en disco para Discovery 1 al precio de 5.000,— ptas.**

NOVEDADES (compatibles SITI V.3)

Sistema experto de Flores de jardín	**3.500**
Sistema experto de Minerales	**3.500**
APLICACIONES SITI V.3	**3.500**

Agenda+Videos+Contabilidad doméstica+Stocks, etc. (necesita el SITI V.3)

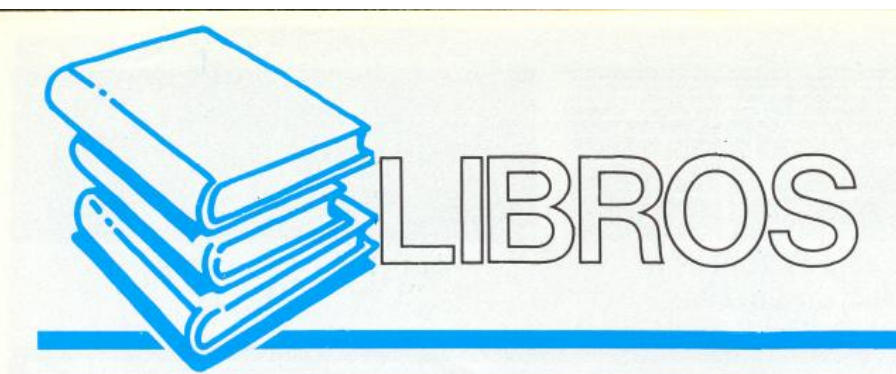
HARDWARE SPECTRUM

Interface sonido TV	**3.500**
Interface Joystik	**2.000**
Joystick Quickshot II	**2.600**
I/F Centronics	**8.000**
Lápiz óptico+software	**4.850**
Interface monitor	**3.900**
Cinta virgen 15'	**100**
Monitor Ciaegi fósforo verde	**24.000**
Monitor Ciaegi fósforo ámbar	**24.750**
Caja-estuche para 12 microdrives	**100**
Teclado Saga 1	**11.000**
Teclado Saga 3	**19.900**
Discovery 1 + disco Kit	**55.000**
Diskettes 3 1/2	**714**
Cable impresora Discovery	**3.500**
Alimentación ininterrumpida	**9.750**
Digitalizador de imágenes P-1024	**35.000**
Impresora Riteman F+	**71.900**



TIENDA AL PUBLICO
EN EL CENTRO DE BARCELONA
HORARIO: de 10 h. a 20 h. ININTERRUMPIDO
SABADOS CERRADO

PEDIDOS POR CORREO O TELEFONO
Envíos contra reembolso a toda España
200 ptas. gastos de envío
En tu domicilio en 3-4 días



LIBROS

El Spectrum tal vez sea el ordenador del que exista más bibliografía técnica, ya que podemos encontrar un buen número de libros que tratan, bien de un modo general o monográfico, sobre todas las particularidades de la máquina, su BASIC, organización interna y posibilidades reales. Sin embargo en muchas ocasiones el usuario de este ordenador necesita un libro guía, que incluya de forma concisa los datos técnicos que le permiten resolver algunos de los problemas que se le plantean en cada momento, sin perjuicio de tener y utilizar otros libros que contengan una información más amplia y completa.

Claves para el ZX-Spectrum, pretende ocupar ese vacío existente en la extensa bibliografía que existe sobre el Spectrum. El usuario encontrará en este libro una forma resumida y fácil de localizar todos aquellos datos que necesite en un momento determinado para trabajar con su máquina.

Está pensado para ser una guía útil, fácil de usar y rápida, estructurado en forma de tablas que contienen toda la información sobre las claves del Spectrum. La intención de su autor es esta, hacer un extenso tratado sobre las posibilidades del BASIC y la máquina. Como libro guía está indicado para cualquier usuario, ya que puede ser muy útil, sobre todo como un buen complemento al escaso manual del Spectrum.



Contiene varias secciones estructuradas en forma de tabla, que facilitan su comprensión y búsqueda de los datos técnicos. Cada sección comprende una lista de las instrucciones y funciones BASIC con un comentario corto y ejemplo, los mnemotécnicos del Z80, códigos de error, juegos de caracteres y plantilla de la pantalla del Spectrum, direcciones que incluye el mapa de memoria. Variables del sistema, una sección de lenguaje máquina que incluye unos esquemas bastante útiles sobre la organización interna del Z80, registros, esquema simplificado del Spectrum, juego de instrucciones del Z80, etc. Incluye además una sección de trucos que serán de gran ayuda al usuario para realizar determinadas operaciones, como mejorar la utilización de la pantalla, cassettes y programas en código máquina.

En general la adquisición de este libro puede resultar interesante para todos los usuarios del Spectrum por la utilidad que representa tener recopilado en un sólo libro gran cantidad de datos necesarios para un mejor conocimiento y manejo del ordenador.

Concurso matemático:

el número más largo

En TODOSPECTRUM sabíamos que sería complicado determinar los cinco factores primos del número 2347915960015735617562 9850677. La redacción de la revista estaba dividida en dos grupos: los pesimistas, convencidos de que nadie participaría en una prueba tan dura, y los optimistas, que auguraban una respuesta masiva (algo más de diez concursantes).

Hasta el último momento, los integrantes de la facción pesimista estuvieron a punto de ver cumplidos sus pronósticos. Pero dos horas antes de que finalizara el plazo, Manuel Arana (viejo conocido de los lectores por sus artículos sobre un nuevo sistema operativo para el Spectrum) se personaba en la redacción para entregarnos su particular solución del problema.

Poco después nos llegaba el correo, con nuevos concursantes de última hora. Y ya casi fuera de plazo, cuando nos disponíamos a abandonar nuestro trabajo diario para jugar al «Starquake». Juan Ignacio Perea, agotado por la carrera, nos entregaba en mano su programa, que resultó ser el ganador.

Sólo Manuel y Juan Ignacio determinaron exactamente los cinco factores primos: 79, 5651, 327979, 82347247 y 1947309701. Ambos utilizaron programas en código máquina, pero mientras que el de Manuel Arana tardó tres días, el de Juan Ignacio Perea invirtió únicamente 23 horas y 49 minutos en descomponer el número propuesto en sus factores primos.

Demetrio Segarra, de Valencia, y Manuel Bautista, de Córdoba, lo intentaron en BASIC, pero abandonaron antes de obtener los cinco factores.

Juan José Rivas, de Cantabria, desistió cuando a su programa en código máquina sólo le restaba determinar los dos últimos factores (en lo que hubiera invertido, como él mismo nos contaba, 2 días, 14 horas y 30 minutos).

A todos ellos, así como a Juan Alonso Expósito (que utilizó un procedimiento erróneo) y a quienes lo intentaron pero no se decidieron a enviarnos sus soluciones, les agradecemos su participación y el haber arriesgado la salud de sus Spectrum.

Asimismo, agradecemos la colaboración de **Pin Soft, ABC Soft** y **Anaya Multimedia.**

En un próximo número incluiremos el listado del programa ganador.

DYNAMITE DAN

Mirror Soft
Spectrum 48 K

El malvado Dr. Blitzen y Donna, su ayudante, tienen en su caja fuerte los planos del terrible Megarrayo Psychon. Nuestra arriesgada misión es guiar al valeroso agente de Dynamite Dan por la casa del doctor para robarlos y evitar que domine el mundo.

Para abrir la caja fuerte contamos con ocho cartuchos de dinamita repartidos por las diversas habitaciones. Una vez recogidos no tendremos más que dirigirnos a la puerta blindada, volarla, coger los planos y huir.

Las diversas pantallas están repletas de objetos móviles difíciles de esquivar, lo que hace casi imposible encontrar la dinamita y llegar a la caja fuerte. Además hay que mantener alto un indicador de energía que aparece en la parte inferior de la pantalla, encontrándose dispersos por las habitaciones diversos objetos de puntuación variable.

Al comenzar la carga del programa se nos advierte que paremos la cinta y, para nuestra sorpresa, aparece un curioso menú donde se nos pregunta nada menos que si queremos vidas infinitas o inmunidad contra los obstáculos y otras facilidades semejantes. No cabe duda que nadie podrá quedarse sin acabar el juego y dependerá de la «honradez» de cada uno el elegir la opción acorde con sus habilidades.

El manejo es sumamente fácil y se realiza sólo con tres teclas que son redefinibles. Salta a la vista desde el principio que, con las consabidas variaciones, el planteamiento no es nada original: se trata de un arcade con numerosas pantallas y dificultades en el que prima la rapidez de reflejos y la habilidad para saltar en el momento justo.

Mirror Soft, empresa que comienza su andadura con buenos programas.

Dan, dotado de infinitas vidas, llega a una de las pantallas más espectaculares.



Control: Teclado, joystick.

Jugadores: Uno.

Gráficos: Buenos, aunque no aportan nada nuevo.

Nivel de dificultad: Regulable mediante la utilización de los diversos pokes incluidos en el programa.

Originalidad: Escasa. Los programas de este tipo, cuya popularidad aumentó notablemente tras la aparición del Jet Set Willy, son ya muy numerosos.

Conclusión: Pese a su poca originalidad, si aceptamos el desafío de robar los planos, el juego es muy adictivo.

TOMAHAWK



Pantalla de carga de Tomahawk, realizado por Dave Marshall, autor del famosísimo Fighter Pilot.

ABC Soft
Spectrum 48 K
2.700 ptas.

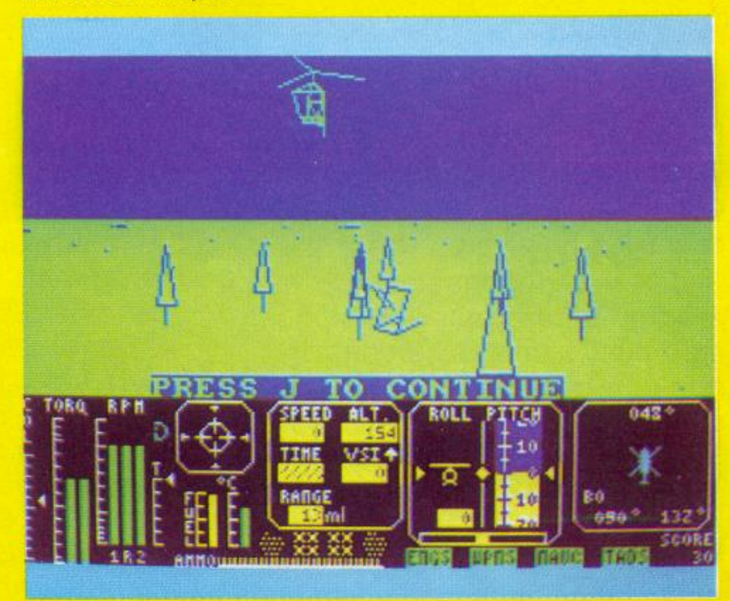
Digital Integration no parece decidida a que las copias de este juego estén al alcance de todos, así que Lenslok se ha encargado de proteger el programa de una forma original, mediante unas lentes de plástico que se adjuntan con la cassette y que colocadas frente al televisor permiten leer el código de seguridad y tener acceso al juego.

Tomahawk es un simulador de vuelo, basado en el helicóptero de ataque avanzado AH-64A Apache de los Estados Unidos. En su diseño intervinieron tanto importantes compañías de fabricación como los mismos pilotos, que dieron sus opiniones sobre las características que debería incorporar. El programa consigue dar una buena sensación de vuelo tanto en el aspecto visual, con gráficos en 3D, como en el sonido, que está adaptado perfectamente, a pesar de la dificultad que supone imitar el ruido del helicóptero.

Al principio puede resultar un poco difícil el manejo de la máquina, pero una vez leídas las instrucciones con detenimiento y después de varios minutos de práctica no hay misión que no se pueda atender. Además, la dificultad puede determinarla uno mismo, eligiendo la misión, las inclemencias del tiempo, volar de día o de noche, etcétera.

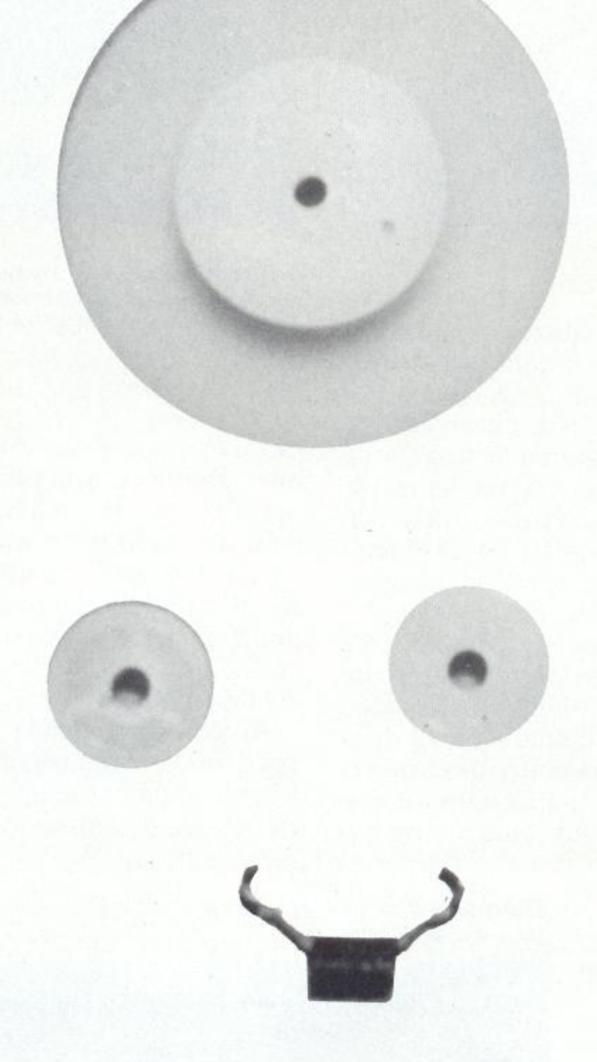
La pantalla se divide en dos partes, una superior más grande donde se observa el terreno, y otra inferior, dedicada al panel de mandos.

El AH-64A Apache dispuesto a lanzarse al ataque.



Control: Teclado, Joystick o doble joystick.
Jugadores: Uno.
Gráficos: Buenos gráficos en tres dimensiones, aunque no resulten demasiado espectaculares.
Sonido: Muy bien adaptado al sonido real de un helicóptero.
Nivel de dificultad: Depende de la elección de las condiciones de vuelo, metereológicas, etc.
Originalidad: La simulación de vuelo con helicópteros no está extendida aún en el Spectrum.
Conclusión: Es un buen programa de simulación de vuelo que no plantea muchas dificultades en su manejo.

De cinta a microcinta



Ciertamente el propietario del *microdrive se acostumbra* tan rápidamente a las ventajas y rapidez de este sistema que evita en todo lo posible acercarse a la «obsoleta» cassette.

Por desgracia, los programas muy raramente se comercializan en *microcinta* por lo que es el propio usuario el que se ve obligado a realizar la conversión, cosa que no es siempre fácil.

Veamos en primer lugar la cinta, ya que hemos de conseguir que el programa tenga sus bloques de código máquina bien definidos y con sus correspondientes cabeceras.

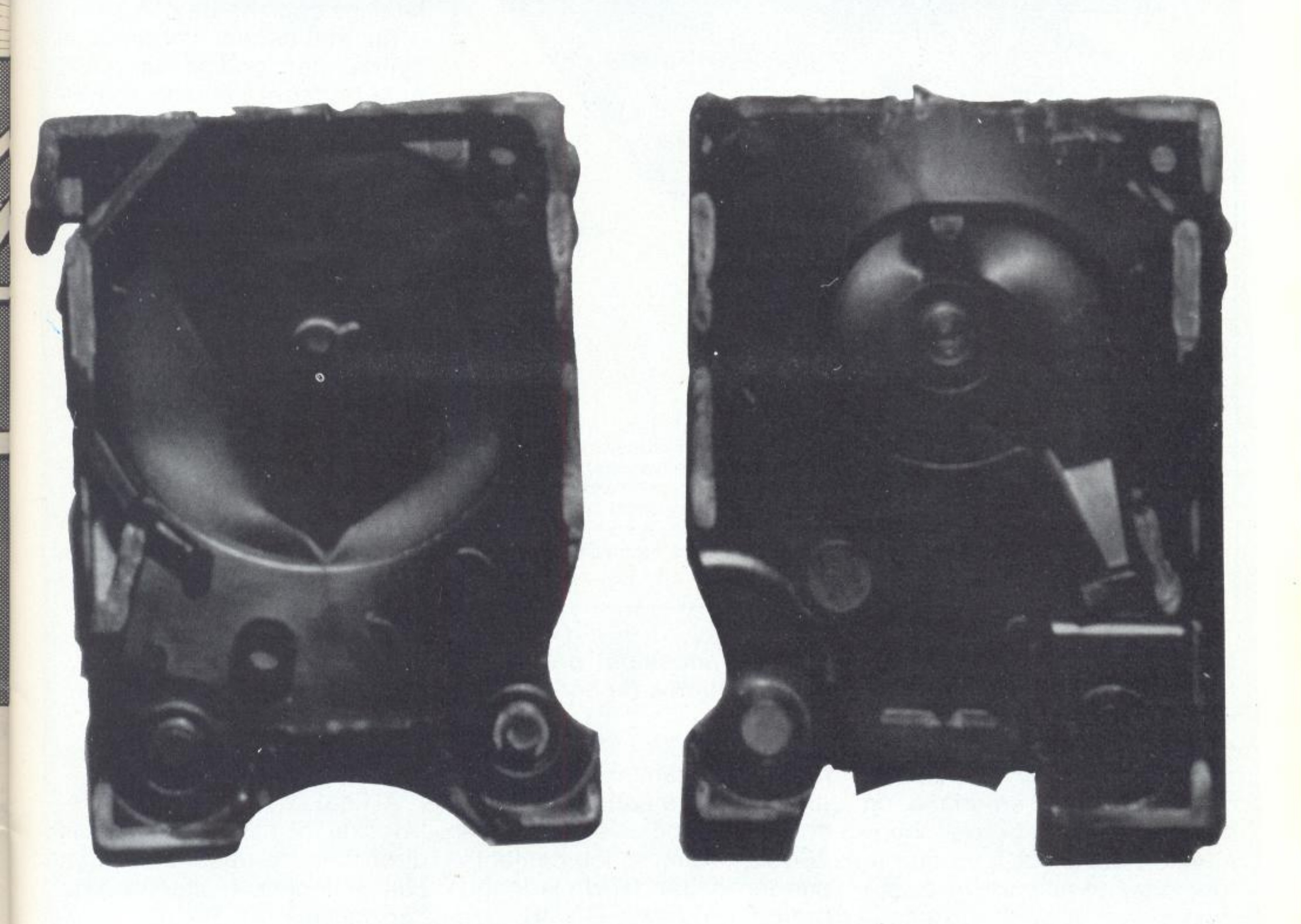
En esta fase nos será de mucha utilidad un programa de los muchos que hay capaces de leer las cabeceras, dándonos los datos imprescindibles de dirección de comienzo y longitud de los distintos bloques de código máquina para salvarlos en el microdrive.

Una vez desensamblado el programa cargador en código máquina, solemos encontrarnos con uno o varios bloques con la siguiente o similar estructura:

```
LD IX,26000
LD DE, 39535
LD A,255 (#FF)
SCF
CALL 1366 (#0556)
```

Lo que significa que el programa arranca en la posición de memoria 26000 (registro IX) y tiene una longitud de 39535 bytes, LD A,# FF es el *flag* indicador del bloque de programa o de datos y por último CALL#0556 es la llamada a la rutina de carga de la ROM. Los datos entre paréntesis son los equivalentes en

Acostumbrarse «a lo bueno» es sencillo. Desacostumbrarse ya no es tan fácil. Nos referimos a las ventajas que supone la utilización de microdrives, raramente empleados en el mercado por su coste superior.
En este artículo, Joaquín Paredes nos demuestra que podemos obviar el problema pasando los programas de cassette a microdrive.



hexadecimal.

De estos bloques podemos encontrar uno o varios según de las partes de que conste el programa; de todas formas al final, casi siempre suele haber una instrucción del tipo:

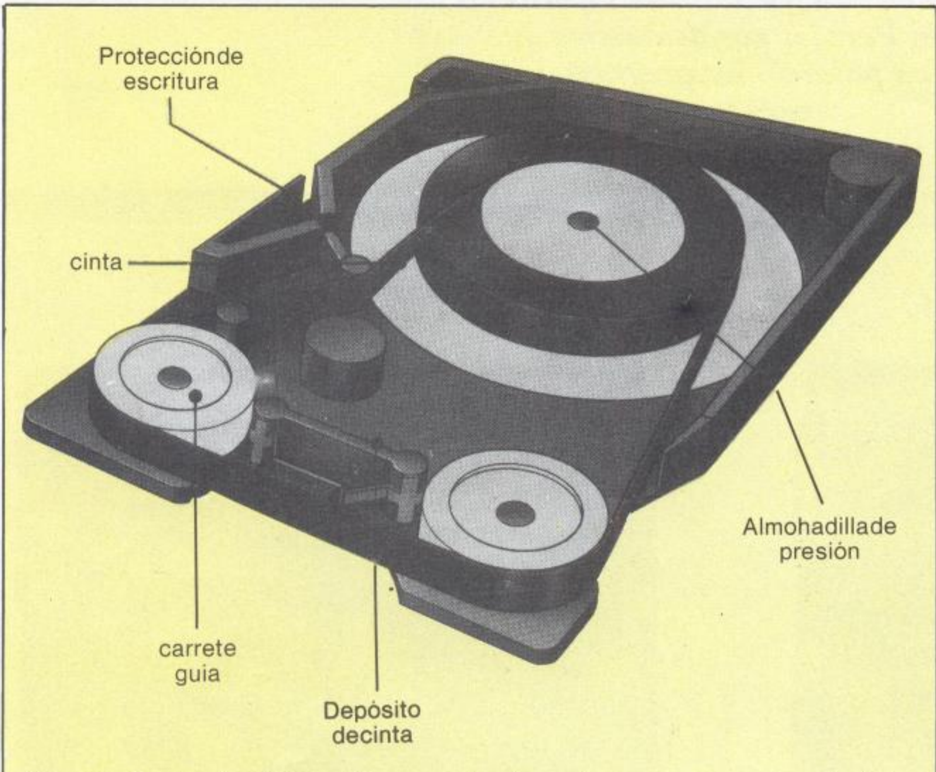
JP 28500

que indica la dirección de ejecución del programa y que equivale a nuestro temido RANDOMIZE USR del BASIC. Todos estos datos nos serán de utilidad para realizar el paso al *microdrive,* como veremos seguidamente.

Cuando realizamos una operación con el *microdrive* dos áreas «crecen» en 32 y 595 *bytes* respectivamente. El mapa contiene información acerca de la distribución de los bloques de 512 bytes del *microdrive.* El *Chanel* tiene un *buffer* de 512 *bytes* que cuando realizamos una operación con el *microdrive* permite el paso de bloques de datos del ordenador al *microdrive* y viceversa. Por si esto fuese poco, hay que añadir otros 58 bytes al grupo de Variables del Sistema.

Así pues, durante una operación con el *microdrive* el sistema necesita 595 + 32 + 58 = 685 *bytes* suplementarios lo que se logra desplazando el programa BASIC y sus variables hacia posiciones más altas de memoria. Evidente-

De cinta a microcinta



Un cartucho de *microdrive* contiene aproximadamente unos cuatro metros de cinta en forma de bucle continuo. Su calidad es superior a la de los cassettes, permitiendo una elevada densidad de almacenamiento de datos. Cuando el *microdrive* entra en funcionamiento, la cabeza de lectura/escritura empuja la cinta contra la almohadilla de presión. Mientras la proteción se encuentre en su lugar, se pueden leer y escribir datos, pero al suprimirla se elimina la posibilidad de escritura, con lo que la información queda totalmente protegida.

mente para programas en BASIC esto no es ningún problema, pero sí para programas en código máquina que ocupen o comiencen en posiciones bajas de memoria o en posiciones altas pero a continuación del BASIC, ya que al funcionar el *microdrive* el código máquina será sobrescrito por el BASIC o incluso por el mapa del *microdrive* o los Canales. También se corrompe siempre el código máquina que se guarda en sentencias REM.

A la vista de todo lo anterior, es evidente que las dificultades del paso a *microdrive* son escalonadas, desde programas en BASIC en donde lo único que tenemos que hacer es pasarlos a la *microcinta* respetando los nuevos comandos, a otros para cuya conversión se necesitan profundos conocimientos de código máquina.

Dentro de la categoría más fácil entran también los de código máquina con direción de comienzo por encima de 25000, aunque lógicamente pueden depender de la longitud del programa BASIC, que puede acortarse en lo posible eliminando comandos de color y también la carga de la pantalla de presentación.

El programa definitivo podría quedar así:

```
10 CLEAR 24999
20 LOAD * "m"; 1; "test"
CODE
30 RANDOMIZE USR (dirección de ejecución)
```

Un problema más interesante es el ocasionado por los programas de código máquina que comienzan en direcciones inferiores a 25000. Estos casos pueden dividirse en dos; los que aún comenzando en direcciones bajas no llegan a ocupar toda la memoria y los que llegan hasta el final, es decir hasta 65535.

En ambos necesitamos la ayuda del código máquina para hacer el traslado de bloques de memoria, ya que no podemos cargar el programa en su lugar correcto directamente desde el *microdrive* por las razones de espacio ya apuntadas.

Para realizar esto utilizamos la potente instrucción de código máquina LDIR, que realiza traslado de datos (a una velocidad vertiginosa) entre distintas posiciones de memoria. Cargamos en el registro HL la dirección de origen del bloque a trasladar, en el registro DE la dirección de destino y en el BC la longitud del bloque:

```
LD HL, org
LD DE, des
LD BC, lon
LDIR
JP eje
```

Al final se ha incluido la instrucción JP (Jump, Salto) a la dirección de ejecucuión, ya que al mover el bloque éste corromperá el BASIC impidiendo la acción de RANDOMIZE USR.

Para los programas que no quieran saber nada del código máquina o no tengan ensamblador, el programa de la figura 1 les solucionará todos los problemas.

El programa solicita el origen del bloque a mover, su destino, la longitud y por último la dirección de su ejecución para realizar la orden de traslado. El programa en código máquina tiene 14 bytes de

largo y se ensambla a continuación del bloque que deseamos mover.

Por ejemplo, un programa que comenzase en 24000 con una longitud de 15000 bytes y con dirección de ejecución en 25500 debemos cargarlo con LOAD ""CODE 3000 y una vez allí cargamos nuestro programa en BASIC y respondemos a las preguntas con los valores de

origen 30000
destino 24000
longitud 15000
ejecución 25500

Seguidamente salvamos el programa en *microdrive* como SAVE *"m"; 1; "test" CODE 3000, 15000+14

Hacemos un RESET, cargamos el programa y lo ejecutamos con la dirección que nos fue indicada, es decir, RANDOMICE USR 45000. No debemos olvidarnos en todos los casos de realizar un CLEAR justo debajo del código máquina que vamos a instalar.

Una dificultad adicional la presentan los programas que comenzando por ejemplo, de 24000 se extienden hasta el límite de la memoria física, es decir, 65535. En estos casos es necesario partir el programa en dos.

Procederemos del siguiente modo: primero cargamos el código máquina y seguidamente pasamos los primeros 200 *bytes* a la pantalla con el bucle de la figura 2, los salvamos momentáneamente en cinta o *microdrive* (CODE 16384, 2000), hacemos sitio para el *microdrive* con CLEAR 25999 y salvamos el resto en *microdrive* con un LOAD *""; 1; "parte 2" CODE 26000, 39535.

Cargamos el programa 1 y sugerimos la línea 270. Seguidamente cargamos los 2000 *bytes* de la pantalla, ejecutamos el programa 1 con GO TO 1 (RUN borraría la pantalla) e introducimos datos que pide (org 16384; des 24000; lon 2000; eje xxxxx). Lo salvamos en *microcinta* como "parte1" CODE 16384,2014 (14 de nuestra rutina) y, una vez con las dos partes, las cargamos y hacemos RANDOMIZE USR 18384 (16384+2000). El programa comenzará a ejecutarse.

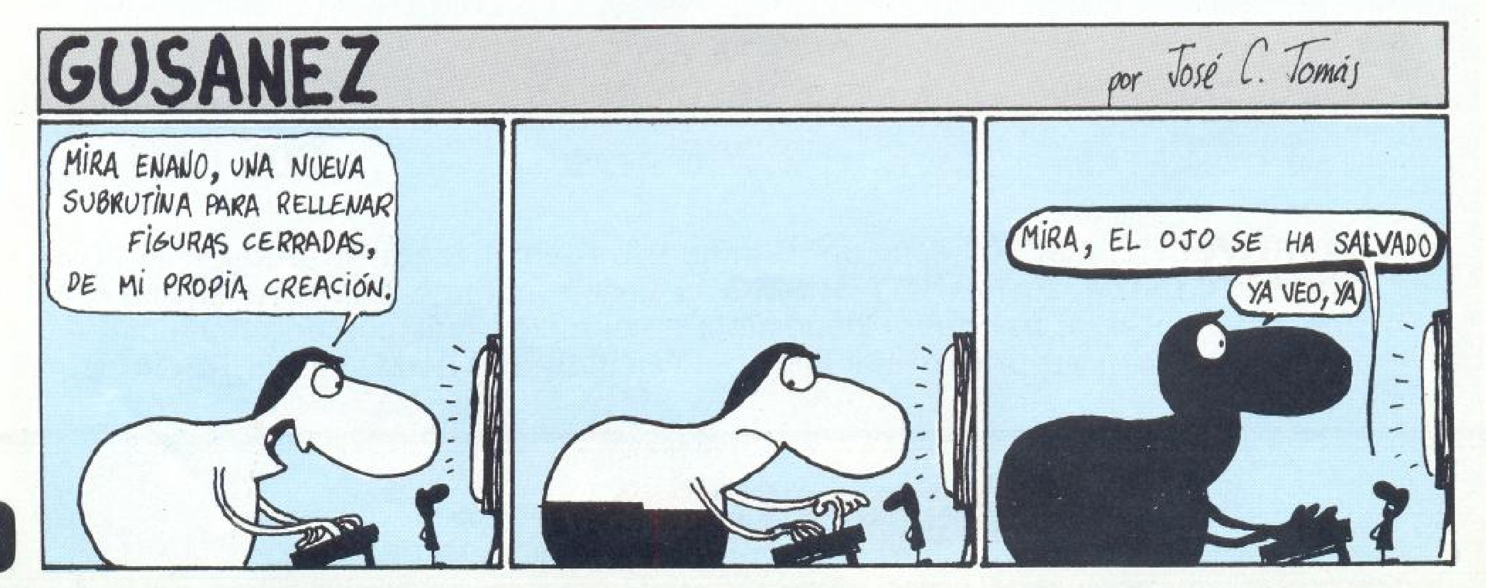
Por supuesto, las características de los programas son tan variadas como programas, hay y este artículo es sólo una mera orientación. Así, otros programas que ocupan toda la memoria es necesario dividirlos con ayuda del código máquina, que por sí solo llenaría otro artículo

Joaquín Paredes Pardo

```
        programa 1
   1 REM =======================
   2 REM
   3 REM Funcion LDIR
   4 REM
   5 REM Joaquin Paredes
   6 REM
   7 REM =======================
   9 GO TO 40
  10 LET b=x-256*INT (x/256)
  20 LET a=INT (x/256)
  30 RETURN
  40 INPUT "origen ";x
  50 GO SUB 10: LET org1=b: LET
org2=a
  60 INPUT "destino ";x
  70 GO SUB 10: LET des1=b: LET
des2=a
  80 INPUT "longitud ";x
  90 GO SUB 10: LET lon1=b: LET
lon2=a
 100 INPUT "ejecucion? ";x
 110 GO SUB 10: LET eje1=b: LET
eje2=a
 200 LET dire=(org1+256*org2)+(l
on1+256*lon2)
 210 FOR c=0 TO 13
 220 READ a
 230 POKE dire+c,a
 240 NEXT c
 250 DATA 33,org1,org2,17,des1,d
es2,1,lon1,lon2,237,176,195,eje1
,eje2
 270 PRINT "PARA EFECTUAR LA ACC
ION DE LDIR HAZ UN     RANDOMIZE
USR ";dire
```

```
        programa 2

 10 FOR v=0 TO 1999
 20 POKE 16384+v,PEEK (24000+v)
 30 NEXT v
```



SERVICIO DE EJEMPLARES ATRASADOS

Complete su colección de

Todospectrum

A continuación le resumimos el contenido de los ejemplares aparecidos hasta ahora.

Núm. 1 • 250 pts.
Cómo usar el microdrive/Programación Basic/Ampliación Basicare/Rutina despertador/Variables del sistema/Entrada datos mediante máscaras/Protección del software/Sintonice su Spectrum/Programas.

Núm. 2 • 250 pts.
Gráficos profesionales/Desplazamiento pixel a pixel/Utilización de rutinas/Construcción del interface centronics/Programas de utilidad para microdrive/Rutina reset en código máquina/Análisis del editor de textos Tasword/Interfaces para impresoras/Programas.

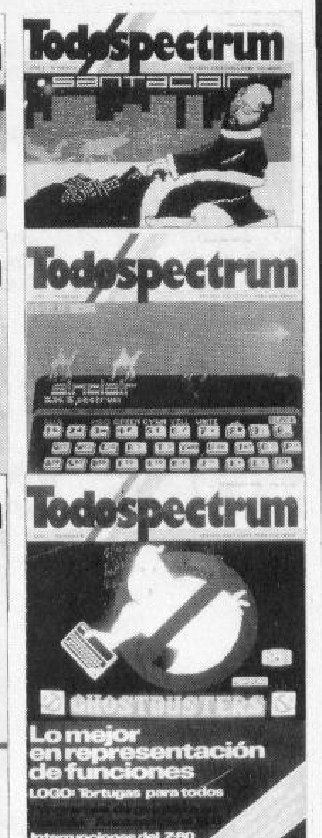
Núm. 3 • 250 pts.
Novedades sonimag '84/Ampliando el Basic/Programas para ordenar programas/Gráficos con el VU-3D/Lenguaje Forth/Archivos en microdrive/Programación de un interface de impresora/Programas.

Núm. 4 • 250 pts.
De profesión: programador/Consola para el Spectrum/Comparación código máquina-Basic/Análisis programa contabilidad /Calendario/Pascal/Programas.

Núm. 5 • 250 pts.
Floppys para Spectrum/Diseño asistido por ordenador/64 Caracteres por línea/Juego de la vida/Pascal/Así hacemos las portadas/Control de evaluaciones/Programas.

Núm. 6 • 250 pts.
Representación de funciones/Todos los caminos conducen a la ROM/Juegos/Pascal/Construcción de un lápiz óptico/Programas de gestión. El SITI/Logo: tortugas para todos/Interrupciones del Z-80/Programas.



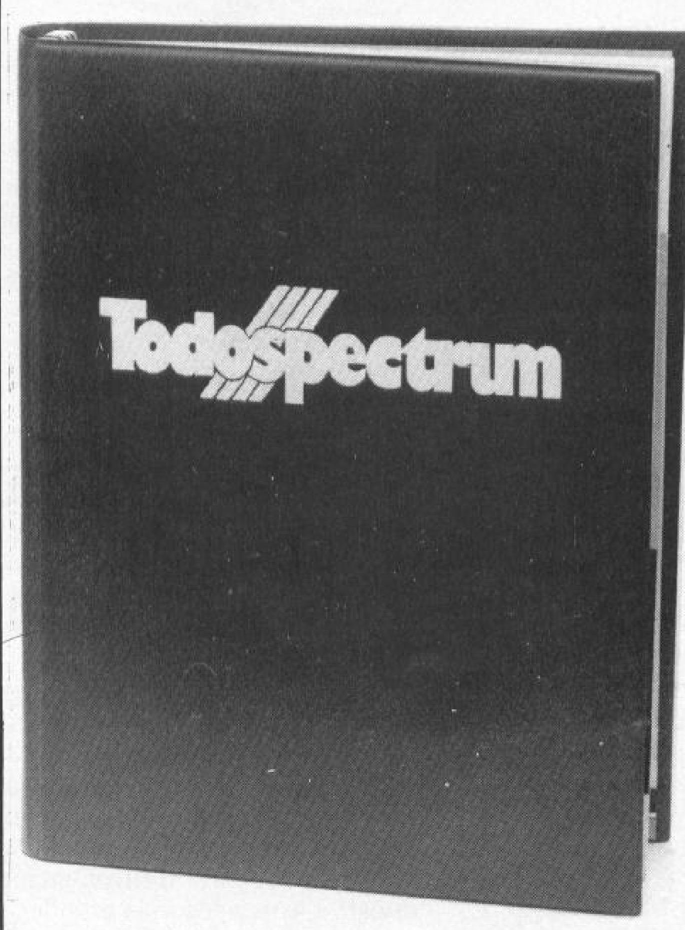


DISPONEMOS DE TAPAS ESPECIALES PARA SUS EJEMPLARES DE



SIN NECESIDAD DE ENCUADERNACION





(cada tapa es para 6 ejemplares)

Para hacer su pedido, rellene este cupón HOY MISMO y envíelo a:

Todospectrum **Bravo Murillo, 377**
Tel. 733 96 62 - 28020 MADRID

Ruego me envíen los siguientes ejemplares atrasados de TODOSPECTRUM .... al precio de 250 pts.

Por favor envíenme .... tapas para la encuadernación de mis ejemplares de TODOSPECTRUM, al precio de 650 pts. más gastos de envío.

El importe lo abonaré
□ POR CHEQUE □ CONTRA REEMBOLSO □ CON MI TARJETA DE CREDITO □ AMERICAN EXPRESS □ VISA □ INTERBANK

Número de mi tarjeta: □□□□□□□□□□□□□□□□□□□□□□□□

Fecha de caducidad .... Firma

NOMBRE ....

DIRECCION ....

CIUDAD .... C. P. ....

PROVINCIA ....

Visión panorámica

de los microprocesadores más comunes

Los microprocesadores son el auténtico motor de los ordenadores. Comúnmente llamados chip, cucaracha y nombres más extraños, siempre se han planteado la duda de cuál es el mejor para un ordenador determinado. El presente artículo, no pretende ser un curso de iniciación rápida a los microprocesadores, sino más bien, una pequeña comparación entre los chips MPU más comúnmente conocidos.

Para responder a la pregunta ¿Qué es un microprocesador? con detalle, serían necesarios varios libros, dada la complejidad y la profusión de tipos actualmente existentes. Sin embargo, la mayoría de los microprocesdores actuales, salvo alguno muy moderno, parten del mismo concepto.

La idea se le ocurrió a Von Neumann, quien sentó las bases de programa y máquina secuencial programable. Se trata de poder disponer de una máquina, que sin «grandes cambios» se pudiera utiizar para resolver tareas de cálculos diferentes. Para poder materializar esta idea es necesario disponer de una máquina que se pueda programar y por otra parte tener un programa, sucesión de órdenes o instrucciones que resuelvan el problema planteado y puede ser interpretado por nuestra máquina programable.

Qué es un microprocesador

Por tanto se desprende de esta breve introducción, que un microprocesador es básicamente una máquina o autómata que es capaz de ejecutar una secuencia de instrucciones, que previamente le ha sido dada o introducida. Esta idea ha sido materializada de diferentes maneras con el paso del tiempo, según la tecnología disponible en cada momento. Se empezó con relés, pasando a válvulas y posteriormente a transistores, circuitos integrados SSI, MSI, LSI hasta llegar a la actualidad a los VLSI y dentro de poco tendremos los **VHSIC** *(Very High Scale Integrated Circuit).*

La configuración interna que posee un ordenador o microprocesador se denomina arquitectura. Las piezas con las que está construido un microprocesador, a nivel conceptual, varían según la máquina que se trate, como veremos posteriormente al comparar los microprocesadores más comunes. Sin embargo, al partir todas del mismo concepto, tienen partes comunes. En primer lugar, puede observarse que disponen de una unidad aritmético-lógica, que es la encargada de efectuar las operaciones del programa, con los operandos que hemos especificado en los mismos. Esta unidad es la que por ejemplo, suma, resta, compara y realiza las operaciones lógicas AND, OR, NOT, etc. Como curiosidad hay que señalr, que generalmente los microprocesadores no suelen disponer de multiplicación o división, salvo los más grandes y modernos, que sí las incorporan.

También disponen los micro-

procesadores de una unidad de control de proceso, que es la que se encarga de buscar la primera instrucción del programa y, a partir de ésta, busca la siguiente al terminar la ejecución de la instrucción en curso. Las instrucciones que «toman» datos de la memoria o «dejan» datos en la misma, son ejecutadas por esta unidad, así como todas aquéllas que tengan realización con el «control» del programa, tales como los saltos.



Estas dos unidades son las que suelen englobarse en lo que suele denominarse *chip* microprocesador o **MPU** *(Micro-Procesador-Unit)*. La denominación **CPU** para estos *chips,* a juicio del que escribe estas líneas, es por del tanto imprecisa al hacer referencia sólo a una parte de lo que contienen.

La tercera unidad que constituyen los ordenadores básicos es la memoria o lugar donde reside el programa o secuencia de instrucciones que nosotros deseamos ejecutar. No nos vamos a extender con las memorias, pero vamos a aprovechar para resaltar el hecho de que como mínimo la **MPU** debe hacer una acceso al ejecutar una instrucción, siendo este acceso o lectura de la memoria precisamente para enterarse de la instrucción que debe ejecutar. Si por ejemplo se trata de una instrucción de suma de dos números, deberá efectuar como mínimo dos accesos más, para buscar ambos números.

En este punto puede introducirse el concepto de longitud de palabra de microprocesador. Si los números que debe sumar nuestra máquina son muy largos, deberá efectuar más accesos a memoria hasta sacar de ésta, todas las «cifras» que componen estos números.

Eléctricamente se pueden representar fácilmente dos estados diferentes, presencia de electricidad y ausencia de ella. Podemos asociarlos con un «1» o un «0» lógicos respectivamente. Si a la vez deseamos representar más estados, podemos agruparlos en grupos de 4 por ejemplo. De esta manera se podrán representar 16 estados diferentes (2 ↑ 4). A cada unidad de este grupo se le denomina *bit (binary digit),* pudiendo adoptar por tanto dos valores. Esos *bits* se van agrupando en potencias de dos, denominándose las agrupaciones de 4 *bits* (2 ↑ 2) «nible», 8 bits (2 ↑ 3) *«byte»* estando menos generalizados, por depender del tipo de microprocesador los de 16 *bits* (2 ↑ 4) *«word»* y para los 32 bits (2 ↑ 5) *«long word».* Resulta claro que a mayor longitud de bits, más posibilidades o estados se pueden representar, siendo por tanto más potentes las máquinas de mayor longitud de palabra y también más cosas. La longitud de la palabra, es por tanto, la unidad de información de los microprocesadores, pudiendo manejar en algunos casos (por ejemplo, el 68020) subunidades o superunidades. En nuestro caso, los microprocesadores de los ordenadores domésticos suelen emplear «todavía» **MPUs** de 8 *bits.* Esto ha sido posible en parte al formato de los chips de memoria, de los cuales una gran variedad están organizadas en 8 bits. Recientemente se ha introducido no obstante un *chip* de memoria organizado en 16 *bits.*

Si se trata por tanto de un microprocesador de 8 *bits* y pudiésemos abrirlo, veríamos paquetes de 8 *bits* en paralelo circulando en su interior.

Su funcionamiento

También serán necearios 8 *bits* conductores para comunicar la **MPU** con la memoria, es decir, el bus de 8 *bits.*

Para realizar las sumas, restas, y demás operaciones se pueden guardar los datos en el interior de la **MPU**. De esta forma si nuestro programa tiene que sumar varias veces el mismo número, en lugar de ir a buscarlo sucesivamente a la memoria, se puede tener guardado este número en un registro interno de la **MPU**, siendo entonces el proceso de suma de nuestro programa más rápido. Para tener una idea de cuánto más rápido, puede

afirmarse que como mínimo es tres o cuatro veces más veloz.

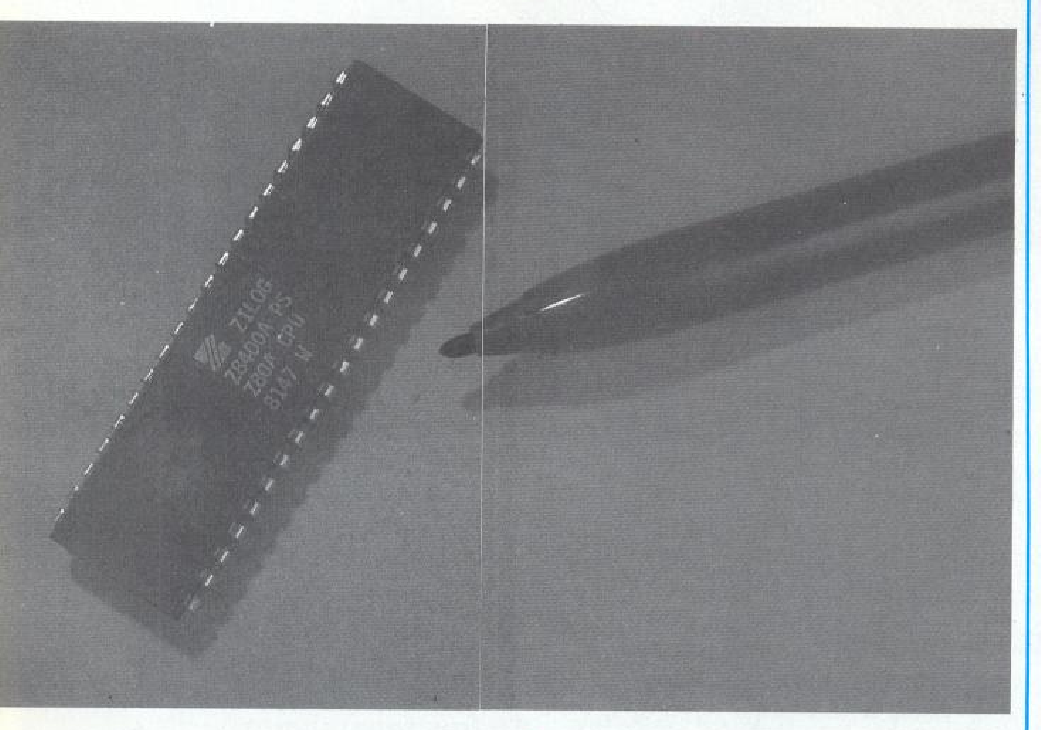
Ahora estamos en condiciones de introducir un nuevo concepto, el de reloj en un microprocesador: la realización de «salto si es cero» en el caso una instrucción, es decir, su ejecución dura cierto tiempo o pasos de reloj. Estos «pasos» están perfectamente determinados para cada instrucción y para cada condición de la **MPU.** Con esto quiero distinguir, por ejemplo, la duración de la introducción de «salto si es cero» en el caso de que la condición sea verdadera o falsa. Si es verdadera, esa instrucción dura unos pasos, mientras que si no lo es, dura otros pasos. Este número de pasos son perfectamente conocidos. Si el reloj da un millón de pasos por segundo, cada paso durará un microsegundo, siendo la frecuencia su inverso, un Megahercio (Mhz).

Entonces si la instrucción de «salto si es cero» dura 5 pasos y 7 si no lo es, nuestro microprocesador tardará en ejecutarla 5 ó 7 microsegundos. Es evidente que cuanto más rápido vaya el reloj, es decir, cuantos más pasos por segundo dé, menos tardarán las instrucciones en ejecutarse. Este concepto puede usarse para comparar las potencias de un microprocesador, viendo cuántos millones de instrucciones por segundo (**MPU**) es capaz de ejecutar, ya que lógicamente la tecnología con la que están construidos los *chips* tienen un techo máximo de frecuencia de reloj, o lo que es lo mismo, una duración mínima para cada paso.

Ahora vamos a comparar algunos de los microprocesadores actuales más comunes. El más popular es el **Z-80,** seguido de 6502 y los menores difundidos en ordenadores personales 6800 y 8085.

Algunos procesadores del mercado

Empecemos por describir los microprocesadores. El 6502 tiene tres registros de 8 *bits,* que son el Acumulador, el Indice X y el Indice Y además del registro de *status-/flags* y dos registros de 16 *bits,* el contador de programa y el puntero de pila o de *stack.* Este último tiene la particularidad de tener el *byte* más significativo siempre a 01, lo que quiere decir en las posiciones 0100 y 01FF tienen que haber RAM. Con relación al juego de instrucciones, hay que señalar que éste no es lineal, teniendo unas instrucciones y unos modos de direccionamiento que no poseen en su totalidad los demás instructores. Además el registro X está algo más favorecido por el juego de instrucciones que el registro Y. En total el **6502** tiene 13 modos de direccionamiento. En lo que respecta a la capacidad de direccionamiento, es la estándar de casi todos los 8 *bits* tradicionales, 64 *Kbytes.* Hay que señalar que los dispositivos de entrada/salida se manejan como memoria, robando espacio vital en ésta. La velocidad de reloj típica del **6502** es de 1-2 Mhz, aunque hay versiones más veloces. Los vectores de interrupción y de reset se encuentran en las posiciones altas de la memoria (FFFA-FFFF) lo que obliga a tener la ROM en estas posiciones. El **6502** puede considerarse como uno de los pioneros, existiendo **MPUs** más modernas derivadas de ella. Como ejemplo, cabe citar *chip* 65C02, que es una versión **CMOS** de la **6502,** pero con importantes mejoras, tales como un juego de instrucciones ampliado, conservando la compatibilidad con la «antigua» **6502.** Muy similar es el **6800,** con dos registros acumuladores A y B de 8 *bits,* registros de status, registro de índice de 16 *bits* y contador de programa, además de un puntero de *stack* también de 16 *bits.* Posee siete modos de direccionamiento y un juego de instrucciones bastante completo, incluyendo instrucciones de *test* de *bits* y complementación, a «1» y a «2». La versión más rápida es de 2 Mhz, incorporando además la posibilidad de conexión de DMA al bus. Brevemente diremos que el DMA es un dispositivo que sin intervención directa de la MPU mueve bloques de datos. Con ello las tranferencias de datos de memoria a memoria o de memoria a dispositivos de entrada/salida son mucho más rápidas. Dentro de la misma familia nos encontramos con otra MPU, la 6809, mucho más moderna que las anteriores, además de más potente, ya que incorpora dos modos de fun-



Son los elementos indispensables en cualquier tipo de ordenador, desde el personal hasta el «mainframe».

cionamiento. Por un lado, el modo usuario y por otro el modo *hardware* o *Supervisor*, con sus respectivos punteros de *stack*. Sobre el **6800** incorpora un registro de páginas de 8 *bits* como uno de 16 *bits*, D. El *stack* de ambos puede ubicarse donde se quiera en el mapa de memoria, siendo más flexible que el **6502.** Los vectores de Reset e interrupción, están en las posiciones más altas de la memoria, debiendo por tanto estar la ROM en estas posiciones, igual que en el **6502.** El **6809** incorpora además la posibilidad de compartir sus buses con otro CPUs iguales o DMA. En cuanto a su juego de instrucciones, posee 13 modos de direccionamiento, además de ser más potente que el del 6800, incorporando la operación del producto sin signo, que no tienen otros CPU's de 8 *bits* tradicionales. También dispone de un modo de escritura del tipo *«early write»* que facilita el uso de memorias dinámicas.

Hay múltiples MPU's en el mercado, cada uno con su característica que le define, pero básicamente son idénticos.

Otra rama de microprocesadores tiene el **8080** en común. De esta CPU se han derivado la **8085** y **Z-80.**Se pueden establecer por tanto una clasificación de **MPUs,** siendo la **6502, 6800** y **8080** de la primera generación y la **65C02, 6809** y **Z-80** de la segunda generación. A caballo entre ambos se encuentran el **8085** más potente al **8080** que al **Z-80.** Por ello, sólo se va a tener en cuenta al **8085** y no al **8080.** El primero de estos dos, dispone seis registros de propósito general de 8 *bits*, el acumulador de 8 *bits* y contador de programa y un puntero de *stack* de 16 *bits*.

Como particularidad está el hecho de poder emplear los registros de propósito general, B, C, D, E, H, L, como registros de 16 *bits*, para efectuar únicamente direccionamientos y sumas, aparte de poderlos incrementar o decrementar. Esta MPU permite también la cesión del bus a dispositivos DMA u otros MPU's contando además con una línea de salida y otra de entrada de datos serie. A nivel de interrupciones esta MPU tiene 5 entradas de interrrupciones, de diferente prioridad, a diferencia de los demás MPU's descritas que sólo disponen de dos entradas. Otra

TERMINOLOGIA

Como todo artículo escrito a alto nivel, puede ser difícil comprender algunas expresiones o terminologías. Por esta razón y sabiendo que nuestros lectores no tienen por que saber lo que algunos términos significan, se ha elaborado la siguiente lista con los posibles términos qué pueden plantear algún problema. Empecemos definiendo lo que es un vector de interrupción. Este es una dirección de memoria, clave o bits que indican la procedencia de la interrupción. Por ejemplo, si el vector 1 está asignado a la impresora y el 2 al diskette, al recibir la MPU el vector 1, pasará a ejecutar el programa de atención a la impresora. Las interrupciones pueden ser de distintas clases y de tipos diversos, en su momento las explicaremos con detalle. Pasemos ahora a definir lo que es un secuenciador programable. Esto es un dispositivo que genera una serie de señales, eléctricas, que permiten el funcionamiento interno de la MPU y otros dispositivos, a nivel de intercambio de señales e información entre los registros internos, buses, rutas de datos del chip en cuestión, en función de la instrucción que se esté ejecutando.

Ahora pasemos a explicar una serie de términos que la mayoría de los lectores conocerán. Se trata de los circuitos SSI, MSI, LSI, VLSI y VHSI. Todas estas iniciales hacen referencia a circuitos integrados diseñados de una forma particular. Los SSI (Single Scale Integration) son circuitos de escala sencilla. Por ejemplo, los chips de puertas lógicas.

MSI (Medium Scale Integration), integración a media escala, que puede ser un decodificador (circuitos diseñados para acceder a direcciones de la memoria).

LSI (Large Scale Integration), integración a muy gran escala, como pueden ser los controladores de memorias dinámicas, memorias EPROM y RAM estáticas hasta 2 K, etc.

VISI (Very Large Scale Integration), integración a muy gran escala. Aquí se engloban los MPU's, periféricos, memorias dinámicas, etc.

Por último, tenemos los circuitos diseñados especialmente para la tecnología del ejército. Se trata de los VHSI (Very High Scale Integration), integración a muy alta escala. En este caso los circuitos son ultrarápidos, que seguramente tendrán pastillas de silicio en tres dimensiones.

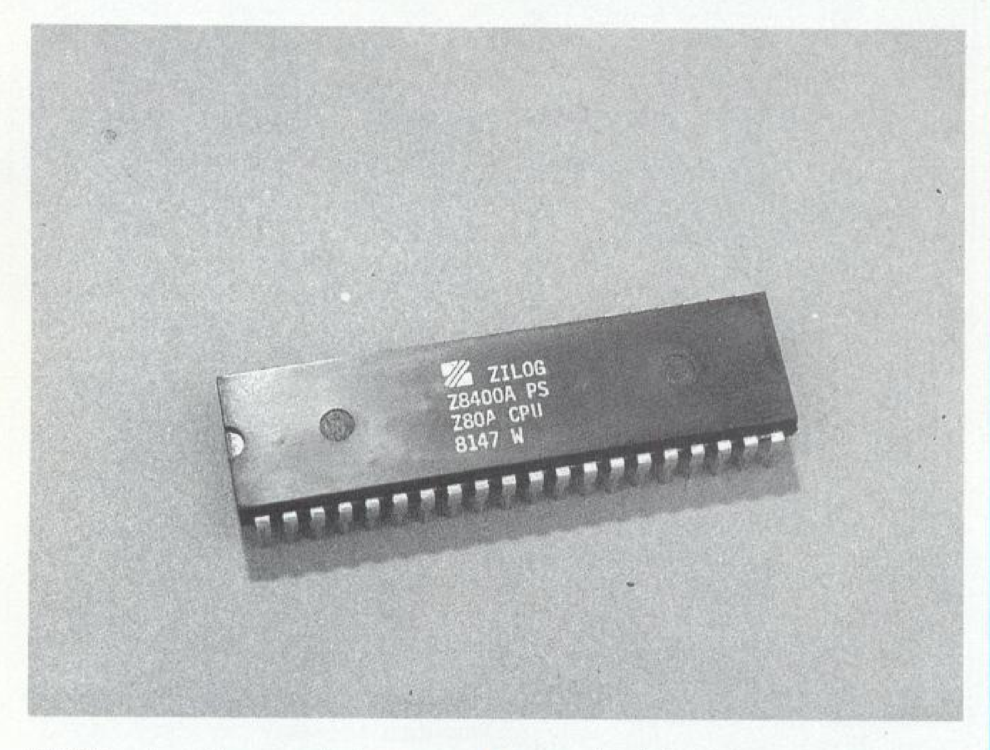
Actualmente se emplean en la electrónica de misiles, guías de tiro y de aviación.

Otros términos que a lo mejor son familiares aunque normalmente no se saben utilizar son los acumuladores y los registros índice. Los primeros son registros en que se realizan las operaciones aritméticas y lógicas en los microprocesadores de 8 bits. En los de 16 y 32 bits, los avanzados y más modernos, tienen la posibilidad de realizar las anteriores operaciones en todos sus registros. En lo que se refiere a los registros índice, éstos son los que permiten ser empleados de propósito general y que además puede ser utilizado como registro de exploración de posiciones de memoria. En él, lo que se guarda son índices que se van a sumar o restar a la dirección del operando.

Aquí finaliza nuestro pequeño glosario de términos interesantes y desconocidos para algunos, aunque para otros no lo sean.

particularidad que también incorpora el **Z-80,** es la división en dos mapas de memoria y de entrada-/salida. El primero es de 16 *bits,* con capacidad de direccionamiento de 64 Kbytes y el segundo permite la conexión de hasta 256 periféricos. Esto ofrece la ventaja de contar con 64 Kbytes de RAM/ROM, incorporando además cualquier periférico paralelo, como interfaces paralelo, serie, controladores de discos, de pantallas, de teclado, generadores de audio programable, etc. En las

Los hay de 8 y 16 bits, todo depende de las necesidades del usuario, pequeños ordenadores de 8 bits y equipos más potentes con procesadores de 16 bits.



MPUs describas anteriormente, cada periférico robaba espacio de memoria, mientras que para ampliarla, se debe recurrir a la técnica de la conmutación de bancos de memoria. La **MPU 8085,** cuenta con una característica única a nivel de bus, están multiplexados el bus de datos (D0-D7) con el de direcciones (A0-A7) por lo que la pastilla dispone para ambos de 8 pines únicamente (AD0-AD7, A8-A15). Junto con el pin ALE se puede distinguir cuando hay datos o direcciones por el bus. Esta técnica también se emplea en MPU's de 16 bits. El juego de instrucciones es amplio, disponiendo de sólo cuatro modos de direccionamiento. Mediante dos instrucciones es posible transmitir o recibir *bit a bit* por el spot serie que incorpora, pudiendo además enmascarar las diferentes interrupciones según las necesidades del usuario. Hay que resaltar que estos **MPUs,** incluida la **Z-80,** dividen por dos la frecuencia del reloj, por tanto si la frecuencia para el **8085** es de 6 Mhz, operará ínternamente a 3 Mhz.

Cómo es el Z-80

Por fin hemos llegado a la **MPU** más popular, el **Z-80.** Incorpora además de los registros del **8085,** un duplicado de todos ellos, incluido acumulador A y registro de *flags* F, excluyendo al contador de programas, que es único e igualmente puntero de pila o *stack.* Este duplicado de los 8 registros A, F, B, C, D, E, H y L, no pueden usarse simultáneamente con los registros originales, sino sólo como sustitutos de ellos. En cambio se realiza mediante la instrucción EX. Esto permite un cambio de contexto muy rápido, sin necesidad de pasar por el *stack o* pila. Además, dispone de dos registros adicionales de 16 *bits,* índice X e índice Y y otros dos de 8 *bits;* R, de refresco para memorias dinámicas y el registro I que refleja el vector de interruptor del dispositivo que la ha provocado. El mecanismo de interrupción es diferente al **8085.** La interrupción vectorizada la posee también el **8085,** provocando un salto a la dirección de memoria

La carrera tecnológica continúa, el próximo paso serán procesadores de 16/32 bits y simplicidad interior de los ordenadores personales, debido a la integración de diversos chips en uno solo.

que indica el periférico que efectúa la interrupción. En el **Z-80,** este vector queda recogido en el registro a tal efecto, siendo el programa el que actúa en consecuencia. Además, permite la programación de tres modos de interrupción, este último, de periféricos **Z-80,** otro compatible con el mecanismo de interrupción vectorizada del **8085** y un tercero para periféricos que no sean de la familia **Z-80.** Dispone además de las líneas para implementar un dispositivo de prioridades de interrupciones del tipo *«daisy-chain».* El juego de instrucciones es claramente más potente que el del **8085,** con 78 instrucciones frente a 158 instrucciones del primero. Tienen instrucciones tan potentes que permiten mover y comparar bloques de datos con una única instrucción, que junto con las operaciones de lectura/escritura de periféricos, rotaciones de todo tipo y los modos de direccionamiento de las mismas hacen de este juego uno de los más potentes en MPUs de 8 bits.

Francisco Cobián Schroeder



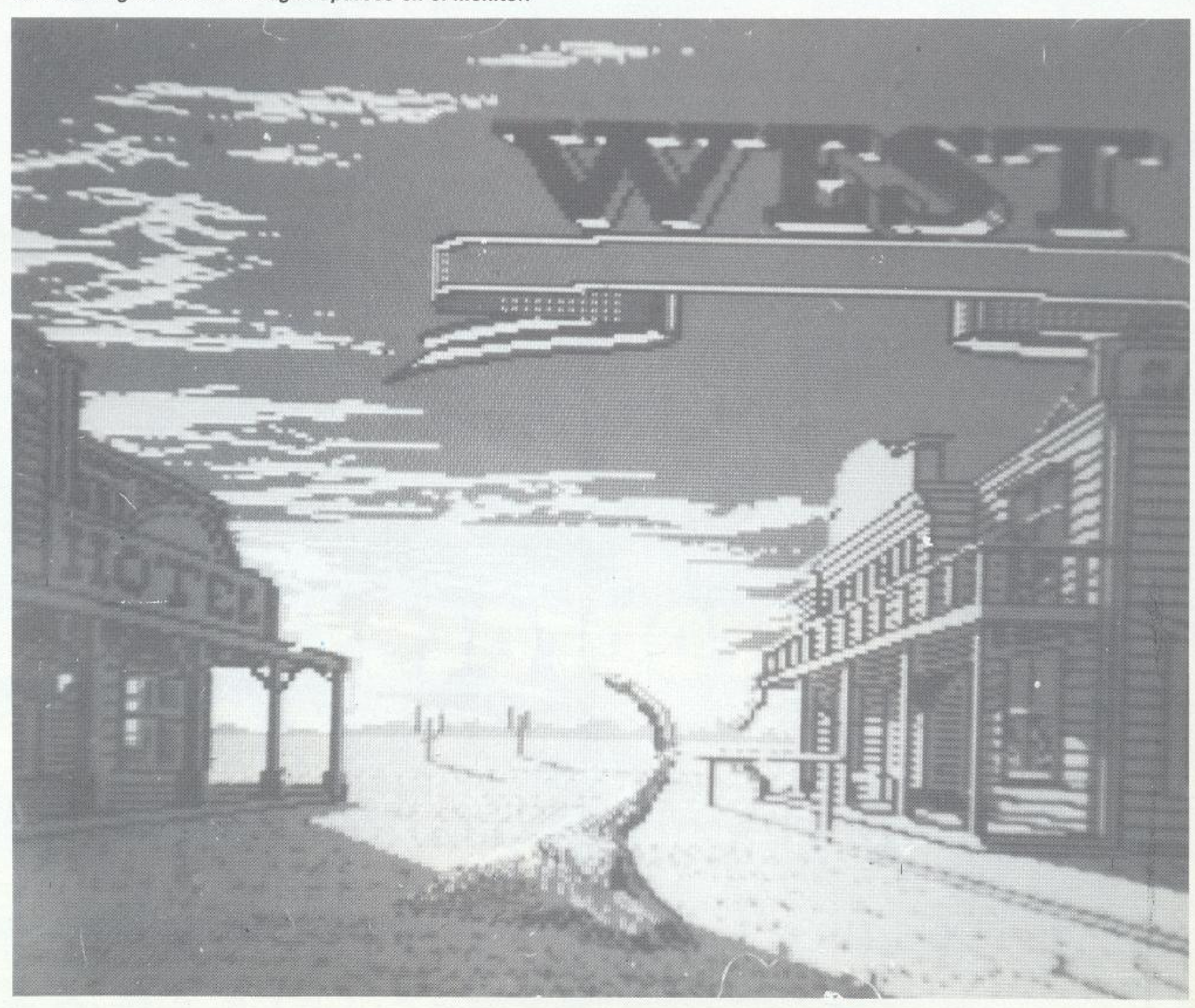
Copy de grises

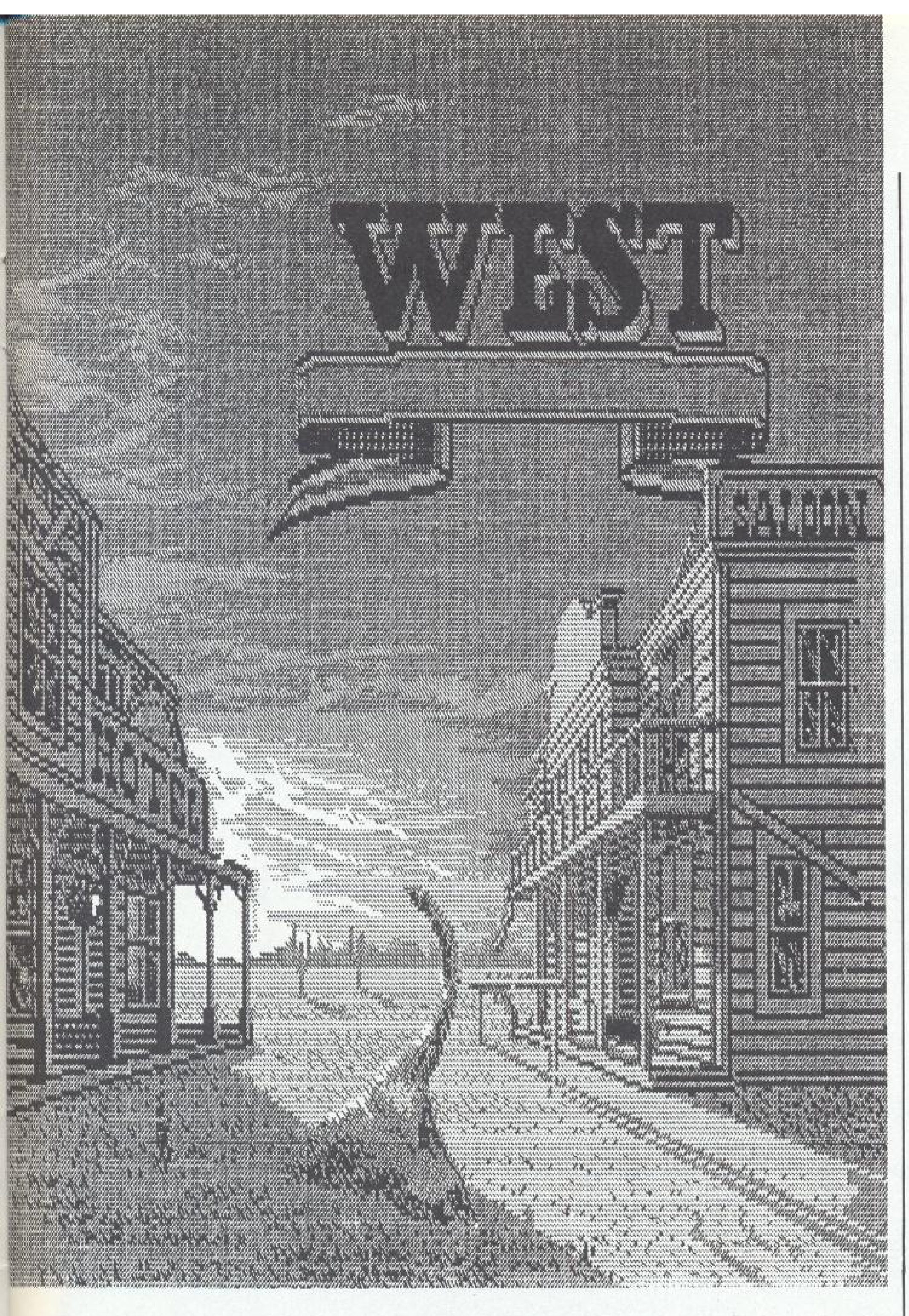
Microprocesadores 68000, una familia numerosa

COPY DE GRISES

Si usted es creador de gráficos, al realizar una copia por impresora de sus dibujos seguramente se habrá sentido decepcionado por los resultados. Con el programa en código máquina que le proponemos en esta ocasión, ese problema no existirá y podrá obtener copias en blanco y negro de forma sencilla con diferentes escalas de grises.

Pantalla original del West según aparece en el monitor.





Copia de pantalla con diversos tonos de grises

El programa «DUMP» es sencillamente un Copy de grises aplicado en esta ocasión al QL. Además de poder simular colores, realiza la copia llenando totalmente la superficie del papel, distingue automáticamente entre modos de alta y baja resolución y permite en cualquier momento, un volcado de pantalla sin tener la necesidad de introducir un comando.

La rutina está preparada para trabajar directamente con una impresora EPSON-FX80 y otras compatibles. Sin embargo, los códigos de control pueden modificarse sin ningún problema para que la rutina pueda utilizarse en otras impresoras de tipo matricial.

Formatos de salida por impresora

El modo *'bit image'* de cuadruple densidad (modo 3) de la impresora FX80 produce 240 puntos por pulgada o 1.920 por línea. La pantalla del QL de baja resolución viene definida por 256 × 256 *pixels.* Pueden obtenerse resultados satisfactorios en la impresora si hacemos equivaler un *pixel* de pantalla con una matriz de 7 × 3 puntos (7 puntos horizontales) en aquélla. Esto produce una imagen de 1792 × 768 puntos o 7,5 × 10,7 pulgadas, con lo cual tenemos cubierta toda la superficie del papel. Precisamente, estos 21 puntos por *pixel* son los que nos van a facilitar la simulación del color mediante una escala de diferentes tonos grises.

Si ejecuta el programa en SUPERBASIC de la figura 1 (ajustando previamente los mandos de color, contrastre y brillo de su televisor), observará que los colores del 0 al 7 siguen una escala ordenada de tonalidades grises. La labor del programa será precisamente la de interpretar estos colores y asignarlos a unas matrices de 7 × 3 puntos (Fig. 2) que representan 7 escalas diferentes de grises desde el negro hasta el blanco, correspondiéndose cada una de ellas con un color diferente. Si piensa cambiar estos modelos de la figura 3 tenga en cuenta que la

APLICACION

impresora Epson no permite colocar puntos adyacentes en el modo 3.

En modo de alta resolución, la pantalla del QL está formada por 256 × 256 *pixels.* En este caso, cada *pixel* de pantalla tendrá que corresponderse con una matriz de 2 × 2 puntos en el modo gráfico CTR (modo 4) de la impresora. Esto produce una copia con unas dimensiones de 6,4 × 14,2 pulgadas. Los cuatro puntos por *pixel* proporcionan cuatro colores (o tonos de gris): negro, verde, rojo y blanco (Fig. 2).

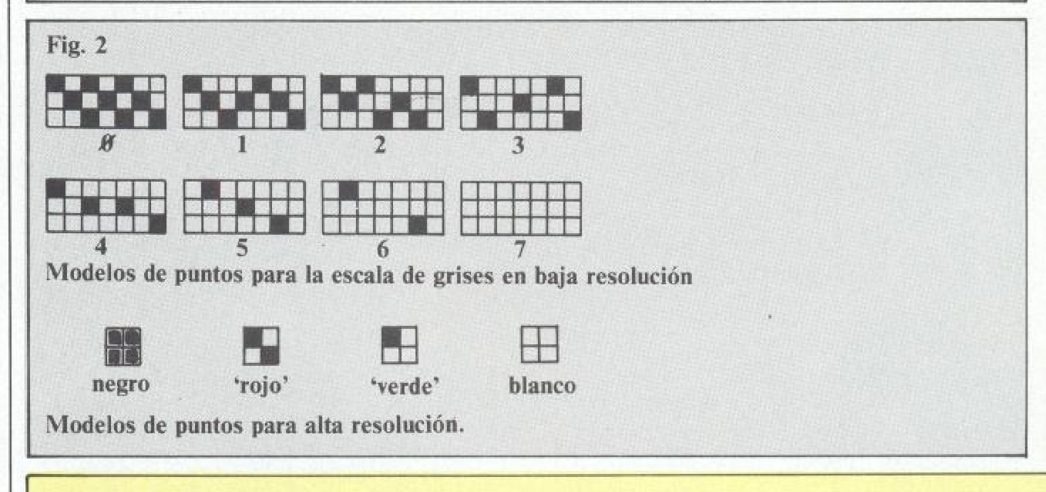
Cómo utilizar la rutina

El código objeto, generado por el ensamblado del código fuente de la figura 3 se guarda en un área de la memoria de la forma acostumbrada con SEXEC para que posteriormente pueda ejecutarse con EXEC y proporciona 50 *bytes* para el *stack* del usuario. Si no está muy seguro de cómo emplear SEXEC, puede examinar el programa cargador de la figura 4, ideado para los usuarios que no dispongan de ensamblador. En este caso, limítese a escribir el programa y hacerlo funcionar. Siguiendo las instrucciones que

Fig. 1

```
1ØØ MODE 8
11Ø WINDOW 512,256,Ø,Ø : PAPER Ø : CLS
12Ø FOR j= Ø  TO 23
13Ø FOR i = Ø TO 7
14Ø PAPER i
15Ø PRINT '      ';
16Ø NEXT i
17Ø PRINT
18Ø NEXT j
```

Fig. 2



Modelos de puntos para la escala de grises en baja resolución

Modelos de puntos para alta resolución.

```
10 MODE 8 : WINDOW 512,256,0,0 : PAPER 2 : INK 7 : CLS
110 AT 9,11 : PRINT 'VOLCADO DE GRISES' : UNDER 1: CURSOR 132,100 : PRINT '
               ' : UNDER 0
120 AT 14,9 : PRINT 'ASEGURESE DE QUE EL CARTUCHO' : AT 16,4 : PRINT 'FORMATEADO
 ESTA EN LA UNIDAD 1'
130 AT 22,9 : PRINT 'PULSE UNA TECLA PARA SEGUIR.'
140 PAUSE
150 CLS
160 AT 10,15 : PRINT 'ESPERE...'
170 RESTORE
180 LET base = RESPR(580)
190 FOR i = 0 TO 11
200 LET sum1 = 0
210 FOR j = 0 TO 49
220 READ a : POKE base+i*50+j,a
230 LET sum1=sum1+a
240 NEXT j
250 READ sum2
260 IF sum1<>sum2 THEN GO TO 330
270 NEXT i
280 SEXEC mdv1_dump,base,580,50
290 CLS
300 AT 10,8 : PRINT 'Utilice: EXEC mdv1_dump' : AT 12,11 : PRINT 'para cargar el
 programa' : AT 14,11 : PRINT 'en codigo maquina.'
310 PAUSE
320 STOP
330 CLS
340 AT 10,6 : PRINT 'Error en datas entre' : AT 12,6 : PRINT 'las lineas ' ;360+
i*60; ' y ';410+i*60;' inclusive.'
350 STOP
360 DATA 96,18,0,0,0,0,74,251,0,10
370 DATA 69,105,103,101,110,95,68,117,109,112
380 DATA 116,1,114,255,112,11,78,65,71,250
390 DATA 0,112,112,17,78,65,8,1,0,2
400 DATA 103,242,71,250,0,106,112,17,78,65
410 DATA 3950
420 DATA 8,1,0,1,103,228,97,0,1,228
430 DATA 65,250,0,96,118,3,114,255,112,1
440 DATA 78,66,74,128,103,6,97,0,1,198
450 DATA 96,202,67,250,0,54,34,136,114,0
460 DATA 116,0,112,11,78,65,114,255,116,255
470 DATA 4507
480 DATA 112,16,78,65,74,1,103,4,97,70
490 DATA 96,4,97,0,0,238,67,250,0,56
500 DATA 97,0,1,82,112,2,78,66,114,0
510 DATA 116,32,112,11,78,65,96,146,0,0
520 DATA 0,0,9,1,0,0,0,0,7,2
530 DATA 2655
540 DATA 9,1,0,0,0,0,0,2,0,3
550 DATA 83,69,82,32,0,2,255,255,0,1
560 DATA 0,1,90,32,0,4,10,27,65,12
```

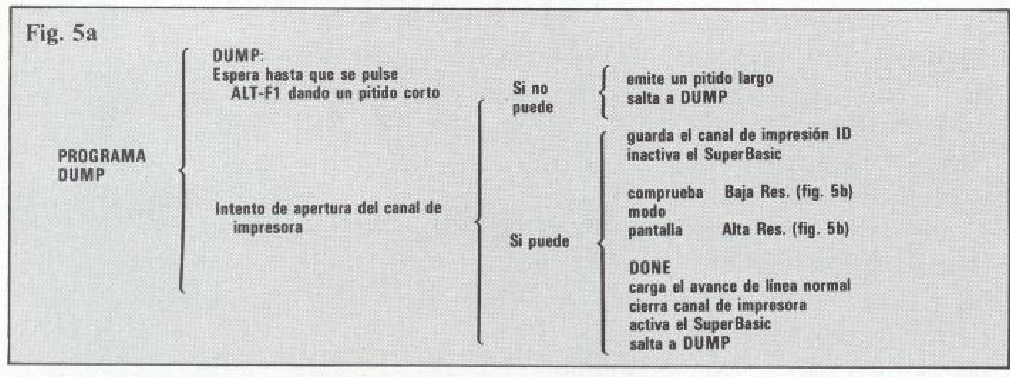


Fig. 5a

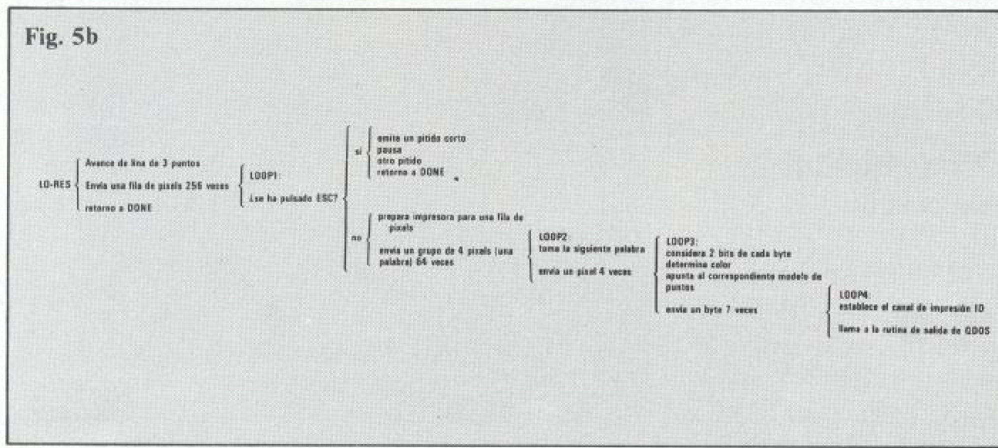


Fig. 5b

```
570 DATA 67,250,0,100,97,0,1,18,42,124
580 DATA 0,2,0,0,62,60,0,255,60,60
590 DATA 2233
600 DATA 0,63,97,0,1,30,67,250,0,66
610 DATA 97,0,0,248,56,29,122,3,112,0
620 DATA 18,4,227,9,227,16,227,9,227,16
630 DATA 227,76,227,16,227,76,231,72,73,250
640 DATA 0,48,216,192,63,4,120,6,18,28
650 DATA 4391
660 DATA 97,0,0,224,81,204,255,248,56,31
670 DATA 81,205,255,212,81,206,255,204,81,207
680 DATA 255,184,78,117,0,5,10,27,90,0
690 DATA 7,0,0,3,27,65,3,0,128,64
700 DATA 160,64,160,64,32,0,128,0,64,0
710 DATA 4718
720 DATA 64,0,32,0,128,64,32,64,128,64
730 DATA 32,0,0,128,0,64,0,32,0,0
740 DATA 128,64,128,32,64,32,0,0,0,128
750 DATA 0,0,0,32,0,0,128,32,0,64
760 DATA 0,128,32,0,0,0,0,0,0,0
770 DATA 1824
780 DATA 0,0,67,250,0,92,97,102,42,124
790 DATA 0,2,0,0,62,60,0,63,60,60
800 DATA 0,7,58,60,0,255,97,112,67,250
810 DATA 0,58,97,76,48,5,239,72,52,53
820 DATA 8,0,112,0,114,8,146,70,227,42
830 DATA 3414
840 DATA 227,16,227,106,227,16,227,72,73,250
850 DATA 0,42,216,192,18,28,97,58,18,20
860 DATA 97,54,81,205,255,216,81,206,255,200
870 DATA 84.77,81,207,255,190,78,117,0,6
880 DATA 10,27,42,4,0,2,0,3,27,65
890 DATA 5055
900 DATA 2,0,192,192,128,0,128,64,0,0
910 DATA 65,250,254,188,32,80,118,255,52,25
920 DATA 112,7,78,67,78,117,65,250,254,172
930 DATA 32,80,118,255,112,5,78,67,78,117
940 DATA 71,250,0,32,112,17,78,65,8,1
950 DATA 4801
960 DATA 0,3,102,2,78,117,97,34,32,60
970 DATA 0,0,255,255,81,200,255,254,97,22
980 DATA 42,95,78,117,9,1,0,0,0,0
990 DATA 1,2,71,250,0,14,112,17,78,65
1000 DATA 78,117,71,250,0,20,96,244,10,8
1010 DATA 3790
1020 DATA 0,0,170,170,50,0,0,0,1,144
1030 DATA 0,0,1,0,10,8,0,0,170,170
1040 DATA 150,140,0,0,0,10,0,0,1,0
1050 DATA 0,0,0,0,0,0,0,0,0,0
1060 DATA 0,0,0,0,0,0,0,0,0,0
1070 DATA 1195
```

aparezcan en pantalla no tendrá ninguna dificultad en obtener el código fuente que necesita.

Para obtener una copia de pantalla, tendrá que cargar el programa en memoria (EXEC mdvl-dump) y seguidamente presionar la tecla ALT y F1 a la vez, con lo cual se oirá un pitido de corta duración. Un pitido más largo le indicará que la impresora está todavía trabajando y no puede realizarse el volcado hasta que haya terminado. La copia de patalla puede interrumpirse en cualquier momento presionando la tecla ESC, entonces dos pitidos cortos le avisarán cuando deba retirar el dedo.

El programa fuente

Antes de utilizar la impresora, «DUMP» proporciona los datos al sistema operativo QDOS para dejar inactivo al SUPERBASIC. Su desconexión no es estrictamente necesaria pero permite hacer una copia de pantalla a una mayor velocidad. El SUPERBASIC se activa de nuevo en conjunción con «DUMP» (tiempo compartido) después de accionar la tecla ESC.



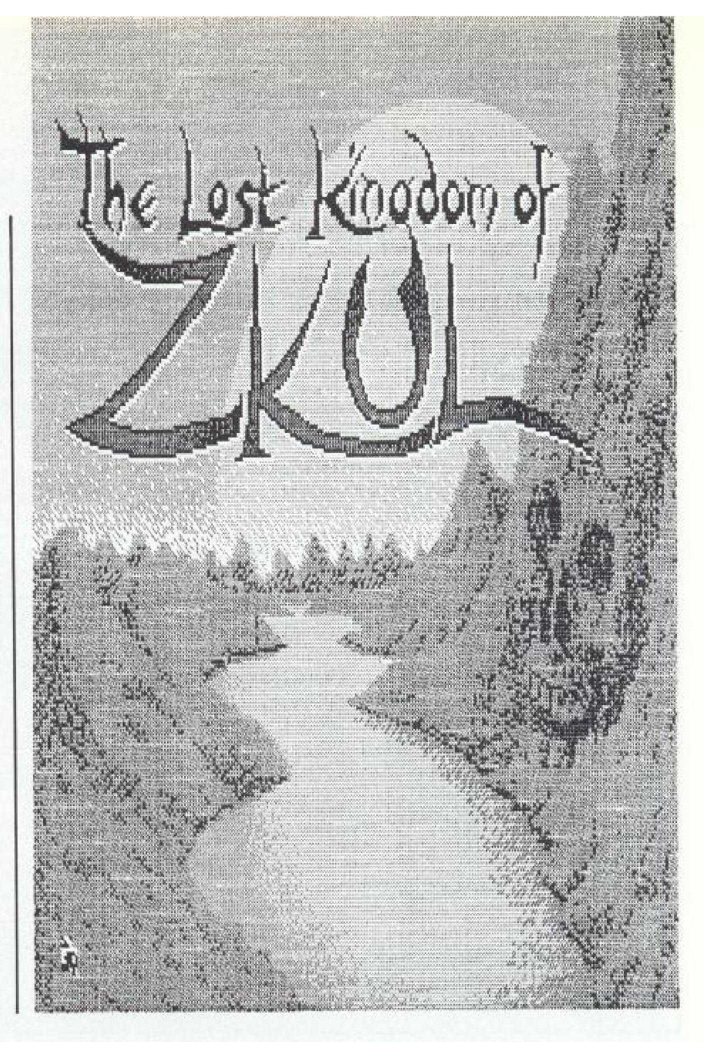
APLICACION

La figura 5 es un esquema general de las subrutinas que emplea el programa en código máquina, el cual queda dividido en tres partes claramente diferenciadas: el programa principal y las rutinas de baja y alta resolución, respectivamente.

Los datos de las subrutinas del programa fuente (Fig. 3) FEED12:, MODE 3:, FEED:3, MODE4:, y FEED2 contienen los códigos de control para la impresora. FEEDx proporciona avances de línea de x puntos. MODE3 tiene la función de dar 240 puntos por pulgada en la impresora y MODE4 asigna 80. Como ya comentábamos anteriormente, estos datos pueden cambiarse para las impresoras que no sean compatibles con la EPSON.

Tratamiento de pixels

La representación de pantalla del QL está formada por 256 hileras de 64 palabras cada una (256 ×




```
* Situa el programa DUMP como tarea independiente:
      GO: BRA      START
          DS.L     1
          DC.W     $4AFB                ESTO ES UNA TAREA
          DC.W     4                    LONGITUD DEL NOMBRE
          DC.W     'Dump'               NOMBRE
* Da al programa DUMP baja prioridad mientras espera ALT-F1
   START: MOVEQ    ^1,D2
          MOVEQ    ^-1,D1
          MOVEQ    ^$B,DO
          TRAP     ^1
* Lee el teclado:
    DUMP: LEA      IPC_ALT(PC),A3       APUNTA AL COMANDO IPC
          MOVEQ    ^$11.DO
          TRAP     ^1
          BTST     ^2,D1                SE HA PULSADO ALT?
          BEQ      DUMP                 SI NO VUELVE A MIRAR
          LEA      IPC_F1(PC),A3        APUNTA A OTRO COMANDO IPC
          MOVEQ    ^$11.DO
          TRAP     ^1
          BTST     ^1,D1                SE ESTA PULSANDO F1?
          BEQ      DUMP
* Reconocimiento de la orden de volcado:
          BSR      BeepShort            PITIDO CORTO
* Intenta abrir el canal de la impresora:
          LEA      PRINTER(PC),AO       APUNTA AL NOMB. DEL CANAL
          MOVEQ    ^3,D3
          MOVEQ    ^-1,D1               CANAL TOMADO POR DUMP
          MOVEQ    ^1,DO
          TRAP     ^2
          TST.L    DO                   ESTABA YA ABIERTO?:
          BEQ      OK                   SI ES ASI CONTINUA
          BSR      BeepLong             SI NO EMITE PIDO LARGO:
          BRA      DUMP                 Y ESPERA OTRA LLAMADA:
      OK: LEA      IDP(PC),A1           ALMACENA
          MOVE.L   AO,(A1)              EL CANAL 10
* Inactiva el SuperBasic:
          MOVEQ    ^0,D2                PRIORIDAD 0
          MOVEQ    ^0,D1
          MOVEQ    ^$B,DO
          TRAP     ^1
* Lee el modo de pantalla del QL:
          MOVEQ    $-1,D2               LECTURA DEL MODO
          MOVEQ    $-1,D1               DE PANTALLA
          MOVEQ    ^$10,DO
          TRAP     ^1
          TST.B    D1                   ALTA O BAJA RESOLUCION?
          BEQ      HIGH
          BSR      LO_RES               VOLCADO DE BAJA RESOLUCION
          BRA      DONE
    HIGH: BRS      HI_RES               VOLCADO DE ALTA RESOLUCION
    DONE: LEA      FEED12(PC),A1        ENVIA ULT. LINEA A IMPRES.
          BRS      PRINT                AVANCE DE LINEAS NORMAL
* Cierra el canal de la impresora:
          MOVEQ    ^2,DO
          TRAP     ^2
* Activa el SuperBasic:
          MOVEQ    ^32,D2               PRIORIDAD 32 AL SUPERBASIC
          MOVEQ    ^0,D1
          MOVEQ    ^$B,DO
          TRAP     ^1
          BRA      DUMP                 ESPERA PARA OTRA LLAMADA
* Almacenamiento para el canal de impresora ID:
     IDP: DS.L     1
```

```
* Comando IPC para la deteccion de la tecla ALT:
  IPC_ALT: DC.B    9,1                  LEE EL TECLADO
           DC.L    0                    4 BITS PARA ENVIAR A IPC
           DC.B    7,2                  ALT EN FILA 7
* Comando IPC para deteccion de F1:
   IPC_F1: DC.B    9,1                  COMO EL ANTERIOR
           DC.L    0
           DC.B    0,2                  F1 EN FILA 0
* Nombre del canal:
  PRINTER: DC.W    3,'SER
           DC.W    2                    DOS PARAMETROS
           DC.W    -1,1                 PORT 1
           DC.W    1,'Z'                ENVIA AVANCES DE LINEA
* Datos para avances de linea normales (12 puntos):
   FEED12: DC.W    4                    ENVIA 4 BYTES
           DC.B    10,27,'A',12
*
* Volcado de baja resolucion:
*
* Avance de linea de 3 puntos:
   LO_RES: LEA     FEED3(PC),A1         APUNTA AL DATO ADECUADO
           BSR     PRINT                LLAMA RUTINA DE IMPRESION
* Proceso de datos de pantalla:
           MOVEA.L ^$20000,A5           COMIENZO DEL ARCHIVO PANT.
           MOVE    ^255,D7              256 FILAS
    LOOP1: MOVE    ^63,D6               64 PALABRAS POR FILA
           BSR     ESC                  SE HA PULSADO ESC?
           LEA     MODE3(PC),A1         PREPARA IMPRESORA PARA
           BSR     PRINT                7x256 BYTES
    LOOP2: MOVE    (A5)+,D4             CARGA UNA PALABRA DE PANT.
           MOVEQ   ^3,D5                4 PIXELS POR PALABRA
    LOOP3: MOVEQ   ^0,DO                LIMPIA EL COLOR
           MOVE.B  D4,D1                * CONSIDERA PALABRA IMPAR:
           LSL.B   ^1,D1                * CAMBIA LOS DOS BITS MAS
           ROXL.B  ^1,DO                * SIGNIFICATIVOS.
           LSL.B   ^1,D1                * ESTOS DETERMINAN LOS
           ROXL.B  ^1,DO                * COMPONENTES ROJO Y AZUL
           LSL     ^1,D4                *
           ROXL.B  ^1,DO
           LSL     ^1,D4
           LSL     ^3,DO
           LEA     File8(PC),A4         APUNTA AL DATO ADECUADO
           ADDA    DO,A4
           MOVE    D4,-(A7)             SALVA PALABRA EN STACK
           MOVEQ   ^6,D4                7 PUNTOS POR PIXEL
    LOOP4: MOVE.B  (A4)+,D1             ENVIA UN BYTE DE DATO
           BSR     BYTE                 A LA IMPRESORA
           DBRA    D4,LOOP4             BYTE SIGUIENTE
           MOVE    (A7)+,D4             RESTABLECE PALABRA
           DBRA    D5,LOOP3             PIXEL SIGUIENTE
           DBRA    D6,LOOP2             PALABRA SIGUIENTE
           DBRA    D7,LOOP1             FILA SIGUIENTE
           RTS                          TRABAJO TERMINADO
* Datos para modo cuadruple densidad de la EPSON:
    MODE3: DC.W    5
           DC.B    10,27,'Z',0,7,0
* Datos para avance de linea de 3 puntos:
    FEED3: DC.W    3
           DC.B    27,'A',3,0
* Modelos para modo de baja resolucion (8 colores):
    File8: DC.B    %10000000,%1000000  NEGRO
           DC.B    %10100000,%1000000
           DC.B    %10100000,%1000000
           DC.B    %100000,0
```

64 × 2/1024 = 32 K). La zona de memoria RAM comprendida entre las direcciones de 2000 Hex. a 27FFF Hex. inclusive contiene la distribución de pantalla de izquierda a derecha, respectivamente.

«DUMP» maneja los *pixels* de distinta forma según el modo de resolución que exista. En copias de baja resolución, las hileras horizontales de la pantalla imprimen a partir de la parte superior de aquélla. En alta resolución se envían columnas verticales de pixels a la impresora, la cual las imprime horizontalmente de izquierda de derecha.

Para que pueda apreciar los efectos de esta interesante rutina, el artículo va acompañado de varias copias impresas en las cuales se observan los diferentes matices de grises que les confieren una gran vistosidad.

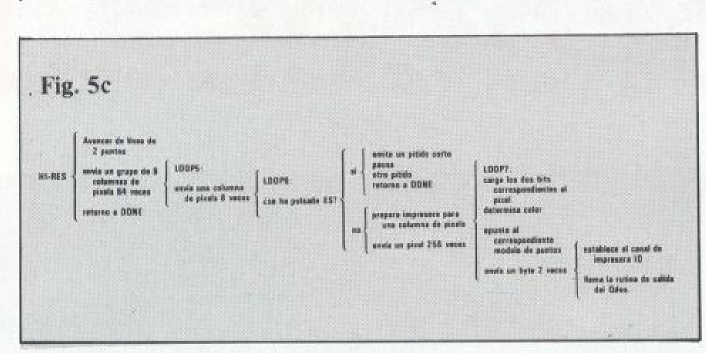
Orlando Araujo Martín

Fig. 7.

Almacenamiento de colores en alta y baja resolución

bit G	bit R	color
0	0	negro
0	1	rojo
1	0	verde
1	1	blanco

bit G	bit R	bit B	color	Código color SuperBasic
0	0	0	negro	0
0	0	1	azul	1
0	1	0	rojo	2
0	1	1	magenta	3
1	0	0	verde	4
1	0	1	cyan	5
1	1	0	amarillo	6
1	1	1	blanco	7

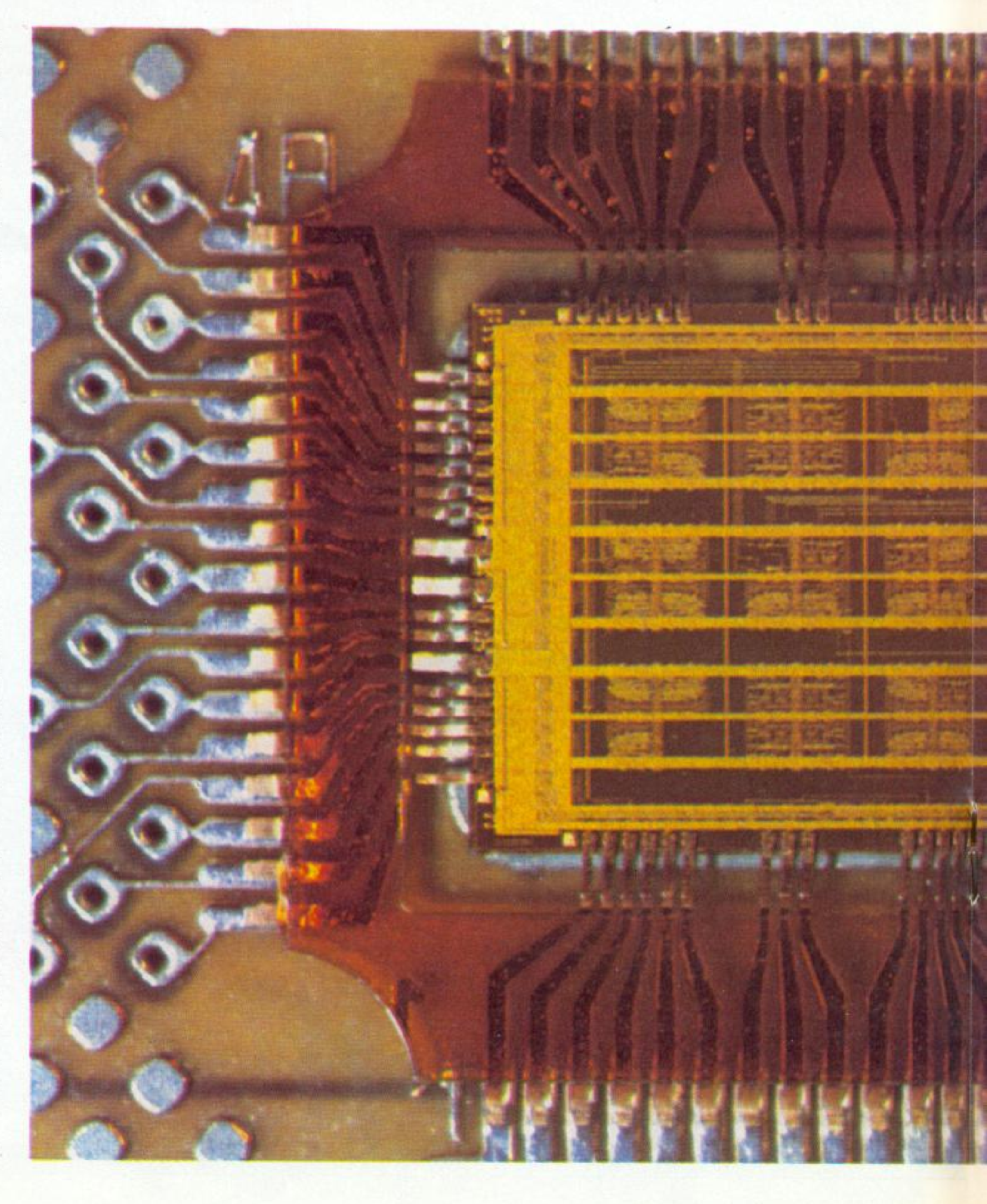



```
          DC.B    %10000000,0            VERDE
          DC.B    %1000000,0
          DC.B    %1000000,0
          DC.B    %100000,0
          DC.B    %10000000,%1000000     AZUL
          DC.B    %100000,%1000000
          DC.B    %10000000,%1000000
          DC.B    %100000,0
          DC.B    0,%10000000            CYAN
          DC.B    0,%1000000
          DC.B    0,%100000
          DC.B    0,0
          DC.B    %10000000,%1000000     ROJO
          DC.B    %10000000,%100000
          DC.B    %1000000,%100000
          DC.B    0,0
          DC.B    0,%10000000            AMARILLO
          DC.B    0,0
          DC.B    0,%100000
          DC.B    0,0
          DC.B    %10000000,%100000      MAGENTA
          DC.B    0,%1000000
          DC.B    0,%10000000
          DC.B    %100000,0
          DC.B    0.0                    BLANCO
          DC.B    0.0
          DC.B    0.0
          DC.B    0.0
*
* Volcado de alta resolucion:
*
* Avance de linea de 2 puntos:
  HI_RES: LEA     FEED2(PC),A1           APUNTA AL DATO ADECUADO
          BSR     PRINT                  LLAMA RUTINA DE IMPRES.
* Procesa los datos de pantalla:
          MOVEA.L ^$20000,A5             COMIENZO ARCH. PANTALLA
          MOVE    ^63,D7                 64 PALABRAS/COLUMNA
   LOOP5: MOVE    ^7,D6                  8 PIXELS POR PALABRA
   LOOP6: MOVE    ^255,D5                256 PIXELS EN UNA COL.
          BSR     ESC                    ESC PULSADA?
          LEA     MODE4(PC),A1           PREPARA LA IMPRESORA
          BSR     PRINT                  PARA 2x256 BYTES
   LOOP7: MOVE    D5,D0                  INDICE DE COLUMNA
          LSL     ^7,D0                  x1282
          MOVE    0(A5),D0.L),D2         CARGA UNA PALABRA
          MOVEQ   ^0,D0                  * SELECCIONA
          MOVEQ   ^8,D1                  * PIXEL
          SUB     D6,D1                  * CORRECTO
          LSL.B   D1,D2                  CAMBIA EL BIT ROJO
          ROXL.B  ^1,D0
          LSL     D1,D2                  Y EL BIT VERDE
          ROXL.B  ^1,D0
          LSL     ^1,D0
          LEA     File4(PC),A4           APUNTA AL DATO ADECUADO
          ADDA    D0,A4
          MOVE.B  (A4)+,D1               ENVIA UN BYTE
          BRS     BYTE                   A LA IMPRESORA
          MOVE.B  (A4)+,D1               Y ENTONCES
          BSR     BYTE                   OTRO BYTE
          DBRA    D5,LOOP7               PIXEL SIGUIENTE
          DBRA    D6,LOOP6               PALABRA SIGUIENTE
          ADDQ    ^2,A5
          DBRA    D7,LOOP5               COLUMNA SIGUIENTE
          RTS                            TRABAJO TERMINADO

* Datos para el modo CTR de la Epson:
    MODE4: DC.W    6
           DC.B    10,27,'*',4,0,2
* Avance de linea de 2 puntos:
    FEED2: DC.W    3
           DC.B    27,'A',2,0
* Modelo de bits para impresora en modo de alta resolucion:
    File4: DC.B    %11000000,%11000000 NEGRO
           DC.B    %10000000,0         VERDE
           DC.B    %10000000,%1000000  ROJO
           DC.B    %0,0                BLANCO
* Rutina para enviar cadenas a la impresora:
    PRINT: LEA     IDP(PC),A0          TOMA LA IMPRESORA
           MOVE.L  (A0),A0             CANAL ID
           MOVEQ   ^-1,D3
           MOVE    (A1)+,D2            N. DE BYTES A ENVIAR
           MOVEQ   ^7,D0
           TRAP    ^3
           RTS
* Rutina para enviar un byte a la impresora:
     BYTE: LEA     IDP(PC),A0          TOMA LA IMPRESORA
           MOVE.L  (A0),A0             CANAL 10
           MOVEQ   ^-1,D3
           MOVEQ   ^5,D0
           TRAP    ^3
           RTS
* Comprueba si se ha pulsado ESC:
      ESC: LEA     IPC_ESC(PC),A3      APUNTA AL COMANDO IPC
           MOVEQ   ^$11,D0
           TRAP    ^1
           BTST    ^3,D1               ESC PULSADA?
           BNE     ABORT               SI ES ASI INTERRUMPE
           RTS
    ABORT: BSR     BeepShort           EMITE
           MOVE.L  ^$FFFF,D0           DOS PITOS CORTOS
    PAUSE: DBRA    D0,PAUSE            COMO RESPUESTA
           BSR     BeepShort
           MOVE.L  (A7)+,A5            CAMBIA ULT. DIR. RETORNO
           RTS
* Comando IPC para deteccion de ESC:
  IPC_ESC: DC.B    9,1                 LEE EL TECLADO
           DC.L    0                   4 BITS PARA IPC
           DC.B    1,2                 ESC EN FILA 1
* Rutina pitido largo:
 BeepLong: LEA     IPClong(PC),A3      APUNTA AL COMANDO IPC
     Beep: MOVEQ   ^$11,D0
           TRAP    ^1
           RTS
* Rutina pitido corto:
BeepShort: LEA     IPCshort(PC),A3     APUNTA AL COMANDO IPC
           BRA     Beep
  IPClong: DC.B    $A,B                ACTIVACION DE SONIDO
           DC.L    $AAAA               8 BITS DE CADA PARAMETRO
           DC.B    50,0
           DC.W    0,400               INTERVALO Y DURACION
           DC.B    0.0
           DC.B    1.0
* Comando IPC para pitido corto:
 IPCshort: DC.B    $A,B                ACTIVACION DE SONIDO
           DC.L    $AAAA
           DC.B    150,140
           DC.W    0,10
           DC.B    0.0
           DC.B    1.0
```

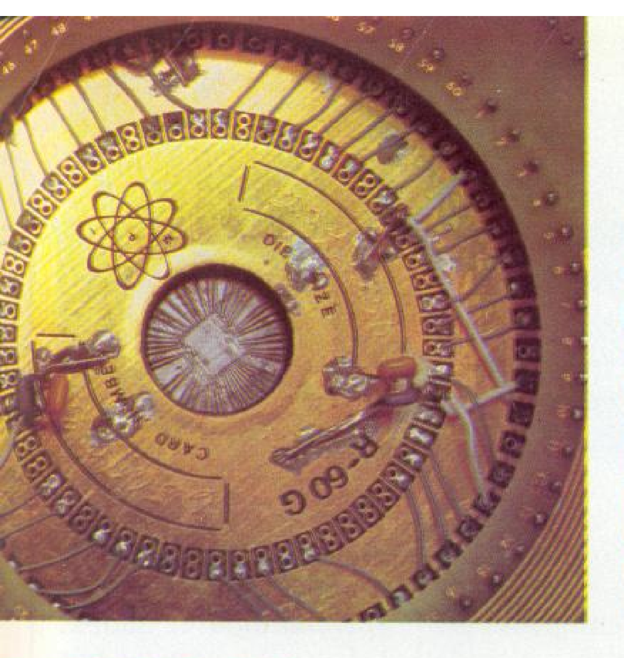
Los 68000, una familia de microprocesado

En estos momentos existen ya ¡cinco! versiones de los procesadores de la familia, y como todos sabemos que el precio de los *chips* desciende continuamente, los que ahora son inasequibles tendrán en el futuro un precio razonable.



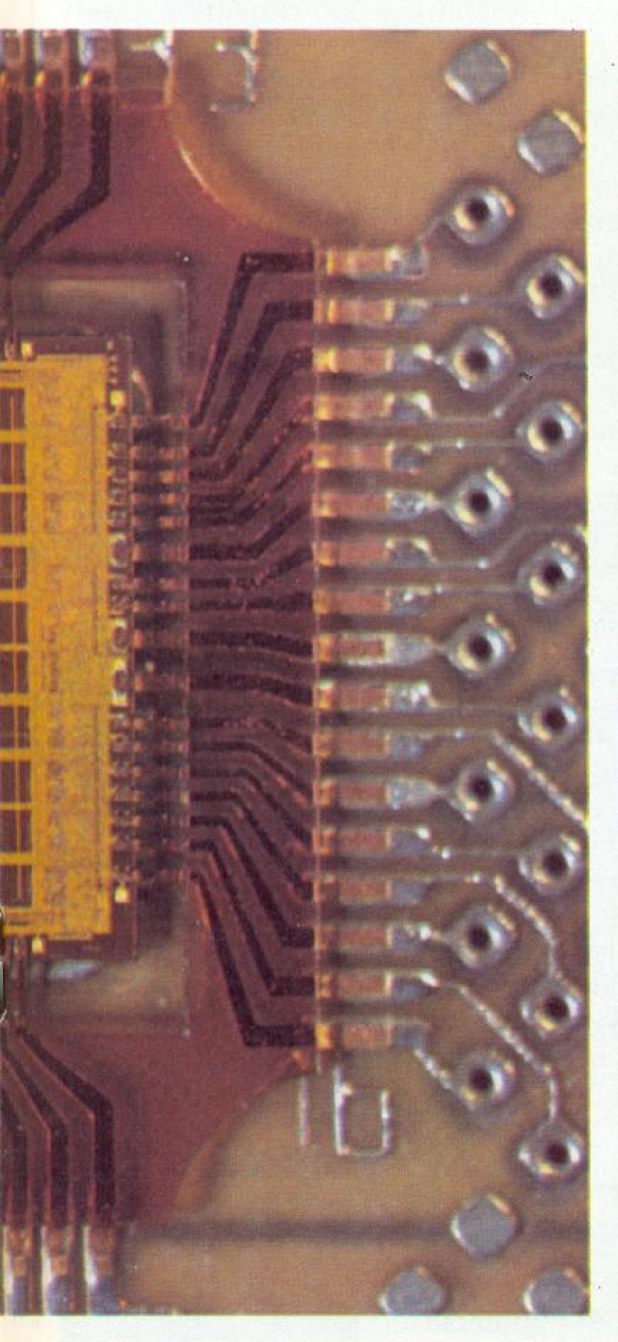
El hermano mayor

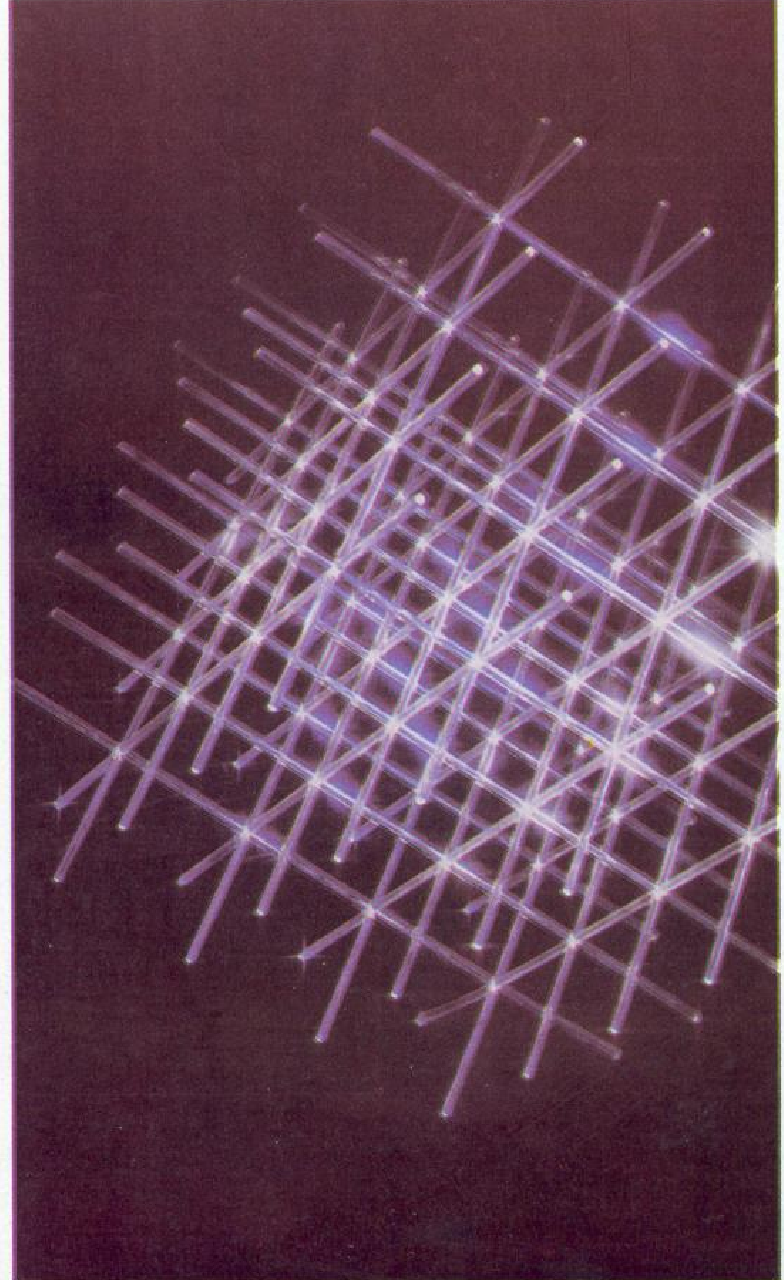
Aunque calificado como un auténtico 32 bits, el 68020 es más que eso: realmente sólo existe otro procesador con la misma potencia, el ATT 32100. Para los más entendidos, los dos tienen una *pipeline* de tres niveles, con 32 *bits* en cada nivel. Esto normalmente permite escribir el resultado de una operación al mismo tiempo que se calcula la siguiente y se toma el dato o la instrucción posterior. Aunque la longitud de la *pipeline*



Los microprocesadores de la familia 68000 surgieron allá por el año 1979. El primero, el 68000, llamado así por contener alrededor de 68.000 transistores, fue el primer microprocesador con arquitectura de 32 bits y estructura interna de 32 btis.
Motorola, el fabricante, desarrolló un procesador totalmente nuevo, incompatible con los anteriores, con una arquitectura basada en los IBM 370 y con una evolución similar a la de éstos.

ores





de estos dos procesadores es inferior a la de otros de 16 *bits,* tiene un tamaño óptimo. Cuando se produce saltos o interrupciones, la *pipeline* se vacía y hay que volver a rellenarla, ocn lo que si es demasiado grande se pierde más tiempo del que se gana.

Los dos tienen la pipelile muy apoyada en *hardware,* con ivarias! alu's de 32 bits. El 68020 tiene una alu para las operaciones, otra para cálculo de direcciones de datos y otra para cálculo de direcciones de instrucciones lo que le permite reconfigurarlas y trabajar sobre 64 bits.

Los dos tienen una *caché* (memoria interna) de instrucciones, donde se almacenan 256 *bytes* (64 palabras de 32 *bits*). Esto les da una gran potencia en operaciones repetitivas, dado que acceden a la vez a la instrucción en la caché y el dato en memoria.

Nacen hacia 1979 y se diseminan por todos los microordenadores de la tierra. Son los 68000, una familia con 5 hijos.

Obviamente tienen coprocesadores numéricos, pero implementados al estilo de los grandes ordenadores y superminiordenadores, como extensiones de la arquitectura y no como periféricos, espacio de direccionamiento de 32 bits, o sea 4 Gygabytes, memoria y máquina virtual, etc. Un detalle fundamental para conocer la diferencia entre estos procesadores y los de 8 y 16 *bits* es que el espacio de direccionamiento es lineal, sin segmentos ni conmutación de bancos, etc., y que este espacio es por tarea o grupo de tareas según el sistema operativo. Manipulan sus 4 Gigabytes con la misma facilidad que nuestro querido Spectrum sus 64 K.

El inconveniente actual es el coste: un 68020 cuesta ilo que un QL antes de la bajada de precio!.

Otra de sus características es la rapidez. E. 68020 tiene tres versiones, con relojes de 12,5 Hz, 16,6 y la anunciada pero no presentada todavía de 24 MHz. Realmente los problemas de velocidad de memorias, jerarquía de éstas, implementación de máquinas virtuales, etc., van a ser muy grandes. La puesta a punto será larga, pero se puede estar seguro que en el Metalab, como en **Apple, Atari, Commodore,** etc., se están «cociendo» ya el 68020.

El 68020 un microprocesador capaz de revolucionar la informática cuando su coste sea más asequible

Hay que destacar una preciosa característica de este procesador: su anchura dinámica de bus, lo que le permite trabajar con ROM, RAM y periféricos tanto de 32 *bits* como de i8 y 16 *bits*! Además los soporta de cualquier velocidad. Esto es extraordinariamente importante para facilitar la portabilidad, no sólo del *software,* sino de los accesorios y periféricos. Hay que comprobarlo, pero resulta muy atractivo, y **Motorola** hasta ahora siempre ha cumplido.

Los virtuales del grupo

Siguiendo para abajo en potencia, tenemos los dos procesadores virtuales: el 68010 y su versión expandida el 68012. Estos ya son de bus de datos de 16 *bits,* y estructura interna similar pero más simple, con pipelines relativamente cortas y con la ventaja respecto a los más pequeños de su modo bucle. Cuando detectan un bucle corto, mantienen las instrucciones dentro del pipeline, accediendo sólo a los datos en memoria, con lo que se consiguen velocidades impresionantes.

La estructura interna es la tradicional de los pequeños procesadores, con

Apple, Atari, Commodore, etc. esperan el 68020 para lograr una mayor potencia en las máquinas y un home computing capaz de sostener aplicaciones fuertes.

tres unidades de cálculo de 16 *bits* trabajando a la vez y con sistemas de comunicación entre ellas. La comunicación entre las unidades de cálculo ha sido mejorada respecto a los más pequeños, acelerándose los cálculos sobre 32 *bits.*

La diferencia entre el 68010 y el 68012 es básicamente que el primero tiene un espacio de direccionamiento de 16 Megabytes y el segundo de 2 Gigabytes.

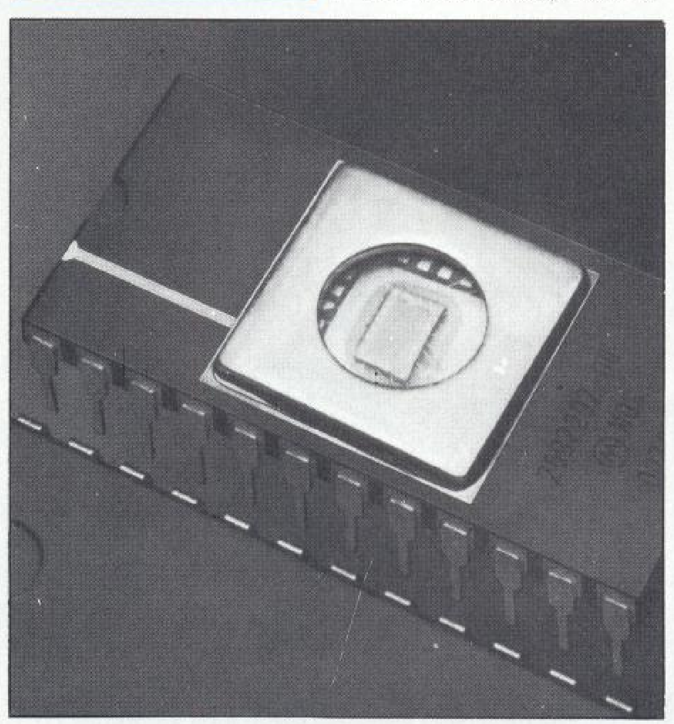
Los más pequeños de los más grandes

Por último, los dos pequeños, el 68000 y el 68008 no soportan máquina virtual, aunque **Apple** en la Lisa consiguió una implementación parcial de la memoria virtual. Su estructura interna es la misma, con tres alu's de 16 bits. La diferencia con los 68010 es la ausencia del modo bucle y una mayor lentitud en operaciones de 32 *bits.*

El 68000 tiene un bus de datos de 8 *bits* bits y un direccionamiento de 16 Megabytes. El 68008, un bus de datos de 8 bits y direccionamiento de 1 Megabyte.

La popularidad de estos micros se está incrementando extraordinariamente.

El 68008, microprocesador del QL, forma parte de los pequeños de la familia, pero su capacidad de direccionamiento de 1 Megabyte le convierten en muy positivo.



Lo más atractivo para los expertos ha sido su similitud con los IBM 370, la arquitectura de mayor éxito en el mundo de los grandes ordenadores. Hasta tal punto son similares, que IBM tiene una versión especial de los 68000, a la que se le ha modificado la ROM interna que determina el juego de instrucciones teniendo estas versiones el juego de instrucciones 370 y los registros 370. Como es obvio, por razones de *copyright,* **Motorola** no ha colocado este juego de instrucciones en sus procesadores de venta libre.

Evolución del mercado

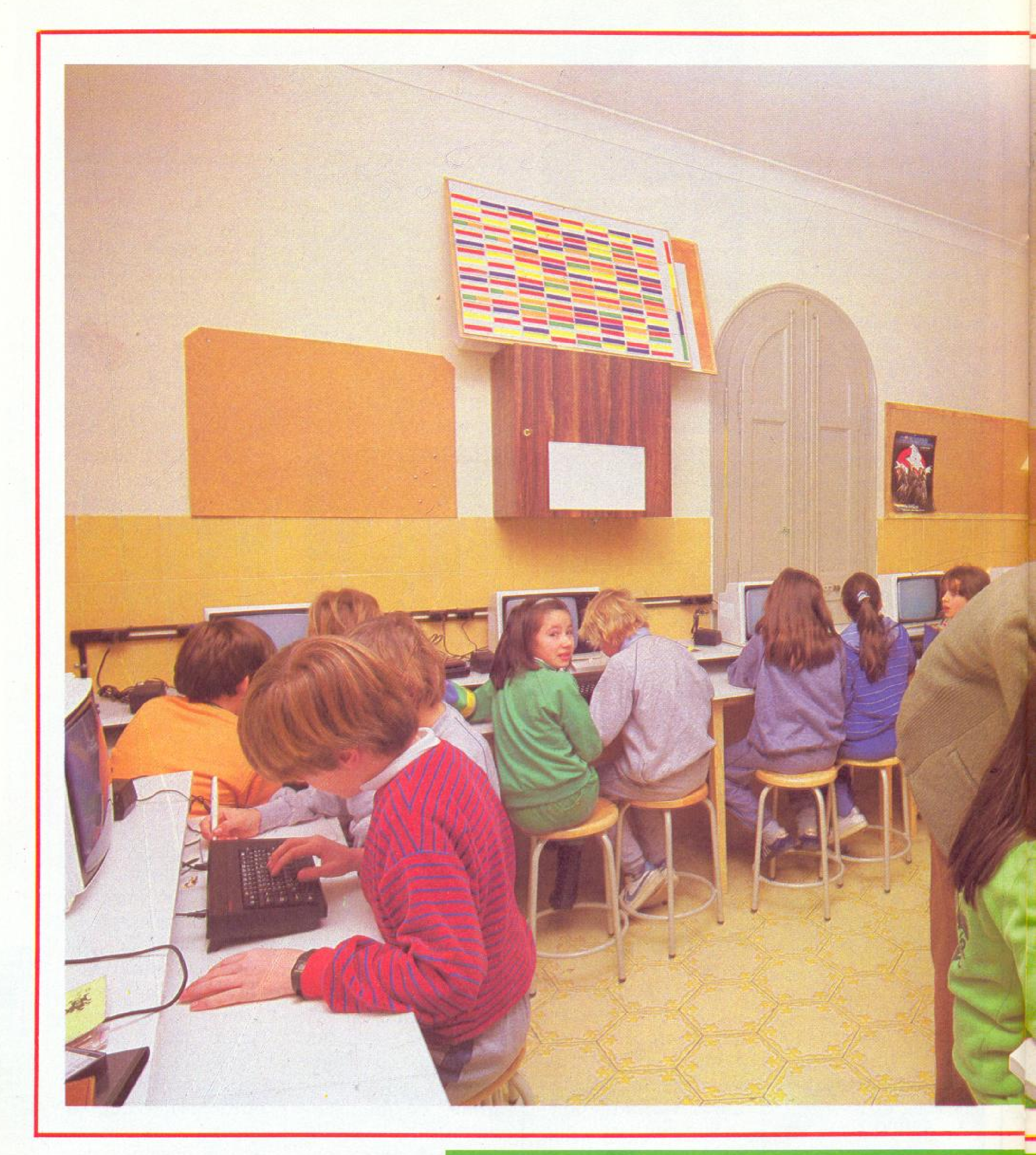
Los 68000 ya tienen una larga historia tras ellos, fundamentalmente como miniordenadores multiusuario, como superpotentes (y supercaras) estaciones de trabajo de ingeniería y como máquinas de inteligencia artificial, disponiendo de aproximadamente una docena de diferentes sistemas operativos. Modelos como el IBM 9000, la serie 200 de **Hewlet-Packard** o los equipos de inteligencia artificial de **Tektronic** son ejemplos de ello; también se utilizan masivamente con el Unix y otros sistema multiusuario.

El **Apple** Macintosh fue el primero en colocarse a niveles de precio de ordenador personal, y el QL el primero en ponerse a tiro de nuestros hispánicos bolsillos, pero va a seguir la marcha con más, mejores y más potentes modelos de todos los fabricantes.

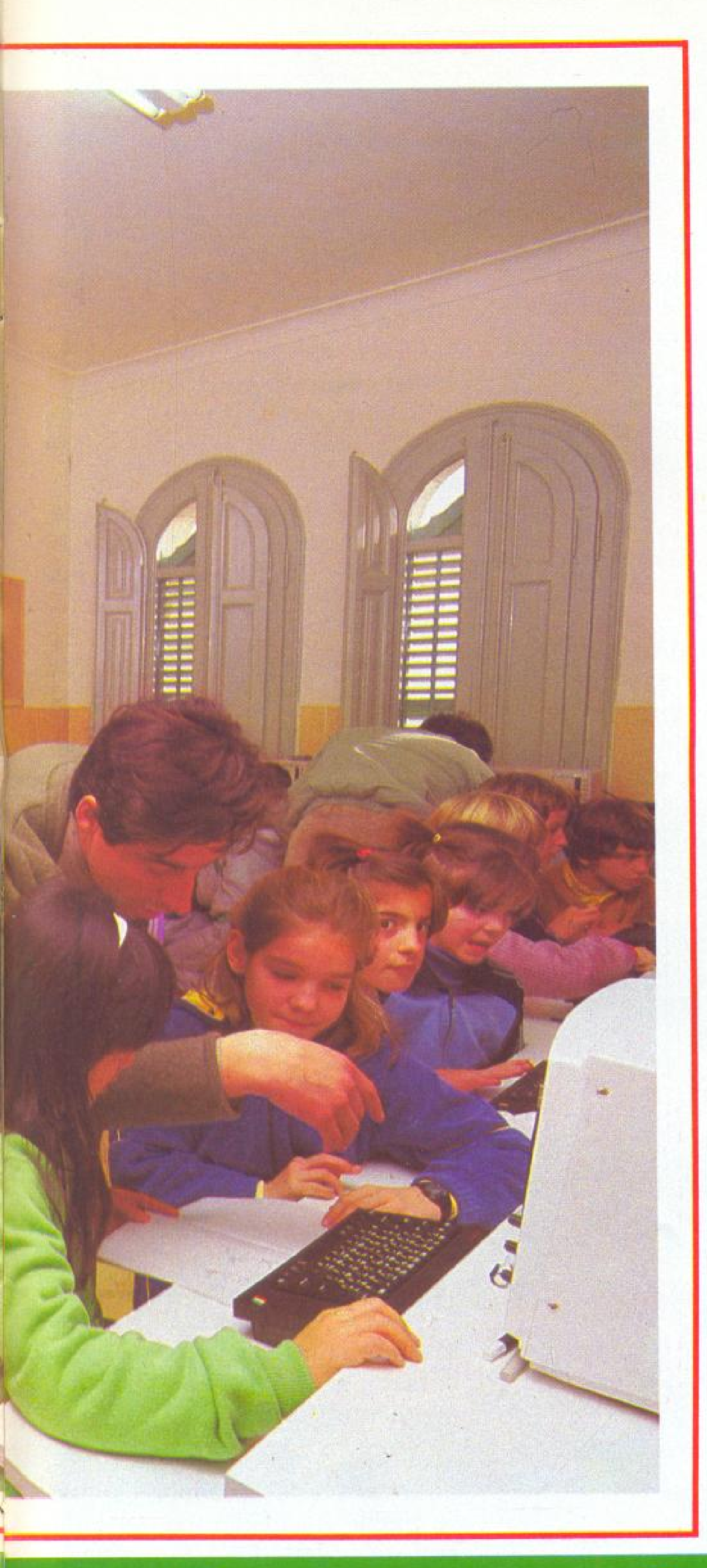
En estos momentos, la carrera de los 68000 en el mercado de micrordenadores de precio asequible no ha hecho más que comenzar. Los modelos que ahora conocemos no son más que los primeros. Va a seguir de momento hasta el 68020, pero dado que los registros de usuario son los mismos en todos, que los mecanismos de direccionamiento son los mismos se tiene garantizada la portabilidad del soft, hasta el 68020 por ahora, claro, que **Motorola** seguirá sacando más procesadores.

Realmente nadie puede evitar que su *hardware* envejezca, pero es necesario poder conservar el *software,* lo más costoso en estos momentos, y la historia de la informática muestra que siempre se ha podido conservar el *software* de los 32 *bits.*

José M. Guzmán



Ordenadores



La introducción del ordenador en la escuela debería haber supuesto un notable avance pedagógico. Sin embargo, en numerosas ocasiones el papel de la informática en el aula ha quedado reducido a su mínima expresión: los cursos de programación en BASIC. Enrique Sánchez analiza a fondo en este artículo las variadas posibilidades del ordenador en el ámbito escolar.

en la escuela ¿Para qué?

Cuando todavía se estaba hablando sobre el gran potencial educativo de la televisión y sobre su posible introducción en las escuelas, todos los niños ya tenían en sus casas un televisor y estaban recibiendo una enorme cantidad de información completamente ajena a los contenidos de los programas escolares. La televisión entró en la escuela, pero no de un modo planificado, sino a través de los niños que vertían en las aulas todo lo que recibían en sus casas; los receptores no llegaron a las clases, pero sí que lo hicieron los mensajes publicitarios, los telefilmes y las noticias que rodeaban a los niños apenas salían a la calle.

Algo similar está sucediendo con los ordenadores: los ordenadores ya han entrado en la escuela aunque todavía no han llegado los aparatos a las aulas.



SITI, uno de los mejores programas de gestión desarrollados para el Spectrum.

Mientras que en los organismos oficiales apenas se ha puesto en marcha un programa nacional para la introducción de la informática y la utilización del ordenador como elemento educativo, muchos alumnos ya tienen un ordenador a su alcance, en su casa, o en la casa de un amigo, o en los comercios en los que pasan largas horas manoseando lo que más tarde o más temprano conseguirán.

Sin embargo, hay una diferencia evidente entre el televisor y el ordenador: nunca se dedicaron tantos esfuerzos a llevar el televisor a las clases como los que ahora se están dedicando a introducir el ordenador. En muchos centros escolares las asociaciones de padres, los claustros de profesores y los propietarios (en el caso de la enseñanza privada) están dedicando abundantes horas a encontrar la manera de llevar hasta su centro estas «máquinas mágicas».

Y las máquinas llegan. Primero es un aparato, luego cuatro o cinco, en el mejor de los casos algunos más, conectados mediante una red local. Los ordenadores son una fuente de prestigio para el centro, una forma de estar a la altura de los tiempos y, en muchas ocasiones, un pretexto para aumentar la cuantía del recibo. Son frecuentes los anuncios de colegios que junto a clases de tenis, danza, inglés en grupos reducidos o judo, incluyen las clases de informática.

Desgraciadamente, los ordenadores no se utilizan o se utilizan mal. La informática, lejos de utilizarse como lo que realmente es, una herramienta, se convierte en mero contenido de los planes académicos y se enseña, generalmente, por personal interesado en el tema pero escasamente cualificado.

La informática está rodeada de una aureola, se ha convertido en un mito —«La informática es la llave del futuro»— y se ha introducido en las escuelas más por la necesidad de no quedarse atrás que por el resultado de un análisis serio.

¿Son tan necesarios los ordenadores en las escuelas? ¿Qué se puede hacer dentro del aula con un ordenador? ¿Qué puede hacer con un ordenador un profesor que apenas conoce los rudimentos de la informática?

El objeto de este artículo es responder a estas preguntas, particularizando en el caso del Spectrum: ¿Cómo se debe utilizar un ordenador en la escuela? ¿Qué es lo que se debe y lo que no se debe hacer? ¿Qué se pue-

de hacer con un Spectrum para potenciar el aprendizaje?

En líneas generales, se pueden distinguir tres grandes campos de aplicación para la informática dentro de la escuela:

— La informática como auxiliar en la gestión y la administración.

— La informática como materia académica de estudio.

— La informática como herramienta didáctica y fuente de recursos.

La informática como auxiliar en el campo de la administración y la gestión

Queda fuera de toda duda, y no es el objeto de este artículo, la gran ayuda, eficacia y ahorro de tiempo que supone la incorporación de la informática a las tareas administrativas. La contabilidad, la facturación de recibos, los ficheros de proveedores, el envío de circulares, etc., son algunas de las múltiples tareas que puede agilizar la utilización de un ordenador.

La informática se ha convertido en un mito —"los ordenadores son la llave del futuro".



Evidentemente, el Spectrum no es el ordenador más indicado para atender a estas funciones (bien escverdad que tampoco está diseñado para ello). Un microordenador del tipo IBM PC o compatibles sería mucho más adecuado para estos propósitos. Ahora bien, existen algunas tareas, tales como el registro sistemático del progreso de los alumnos o la realización de un pequeño fichero que incluya algunos de sus datos pesonales, para las que el Spectrum puede suponer una buena ayuda. Es decir, puede servir como herramienta de apoyo para cada uno de los profesores.

Hay que decir, sin embargo, que generalmente el volumen de datos con el que trabaja un profesor no es lo suficientemente grande como para justificar su inclusión en un ordenador. Para buscar un dato como el teléfono de un alumno, suele ser bastante más rápido recurrir a la típica ficha de cartulina que cargar el programa desde una cinta de cassette y luego acceder a la información.

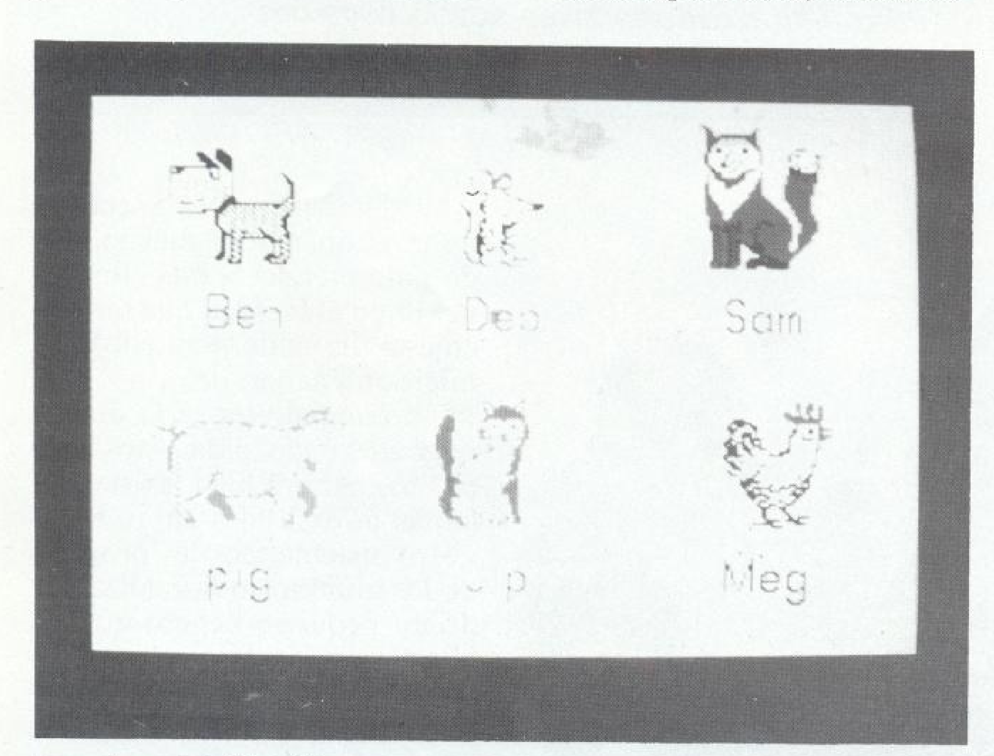
La situación es diferente cuando se trata de seleccionar

una gran cantidad de datos de un mismo fichero. En este caso, la tediosa espera de tres o cuatro minutos queda compensada por la posterior agilidad en la búsqueda, especialmente cuando se realiza una selección como la siguiente: «alumnos que han aprobado la segunda y la tercera evaluación» o «alumnos de tercer curso que necesitan más de media hora para llegar al colegio». Para este tipo de tareas ningún fichero clásico, por muy bien organizado que esté, puede superar a una buena base de datos, tal como el S.I. T.I.

Idéntico razonamiento se puede aplicar a la extracción de datos numéricos. Cualquier tipo de estudio estadístico sencillo se puede realizar mucho mejor y en menos tiempo con una hoja de cálculo que con una calculadora científica normal.

Sin embargo, el empleo de la cinta de cassette como memoria de masa tiene sus inconvenientes: para modificar tan sólo un dato es necesario volver a grabar todo el fichero, con la consecuente pérdida de tiempo y los riesgos que esto conlleva (posibilidad de que la grabación no se realice correctamente y se pierda una parte o la totalidad de la información).

En cualquier caso, existen sobradas ocasiones en las que el Spectrum, incluso en su configuración básica, puede suponer un apoyo útil y agradable.



***Learn to read*, de Macmillan, empresa especializada en software educativo.**

La informática como materia académica de estudio

En la mayoría de los centros escolares la introducción de la informática ha quedado reducida a la inclusión de una asignatura pomposamente llamada Informática, que consiste en realidad en un curso, de más o menos calidad, de programación en lenguaje Basic.

Limitar la enseñanza de la Informática a la enseñanza del Basic es un grave y extendido error. La programación en Basic, o en cualquier otro lenguaje, dista mucho de ser lo más importante en este campo. Sobre todo si se limita a la exposición de un conjunto de trucos y de instrucciones, pero no viene acompañada de la presentación de los soportes lógicos de aquello que se está haciendo.

Hoy en día es perfectamente posible utilizar eficazmente los servicios de los ordenadores sin tener conocimiento de un lenguaje determinado (al igual que es posible utilizar correctamente un automóvil sin tener grandes conocimientos de mecánica o un televisor sin conocer el fundamento físico del tubo de rayos catódicos). Cualquier usuario de un ordenador no puede dedicar horas preciosas a la elaboración de unos programas que personas especializadas ya han realizado mucho mejor que él. Lo que debe hacer es aprender a sacar el máximo partido posible de los programas existentes en el mercado.

En la enseñanza de la Informática debería dedicarse una buena parte del tiempo a la adquisición de una cierta destreza en el manejo de los clásicos paquetes de aplicaciones.

El Spectrum es perfectamente válido para cumplir este objetivo. Entre los programas disponibles en el mercado existen algunas buenas hojas de cálculo, bases de datos, procesadores de textos, diseñadores de gráficos, etc. Su manejo no es excesivamente complicado y sus

prestaciones son muy buenas, sobre todo si se consideran desde un punto de vista didáctico, como paso previo a la utilización de sistemas más complejos.

La informática como herramienta didáctica y fuente de recursos

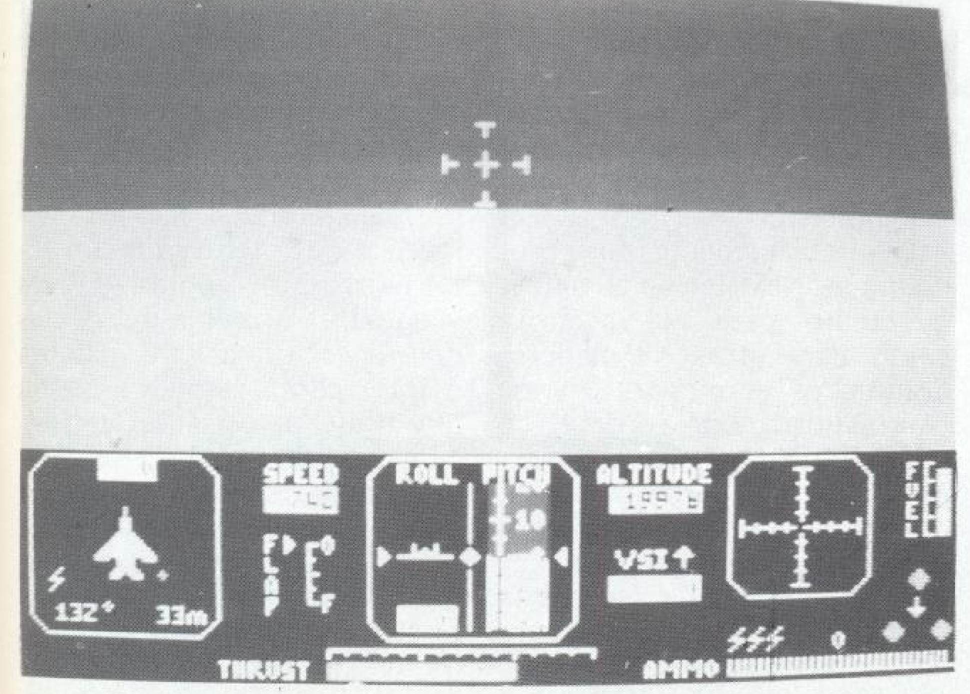
Este es el aspecto que más nos interesa. No podremos decir que hemos incorporado la

> ***Limitar la enseñanza de la informática a los cursos de BASIC es un error grave y extendido.***

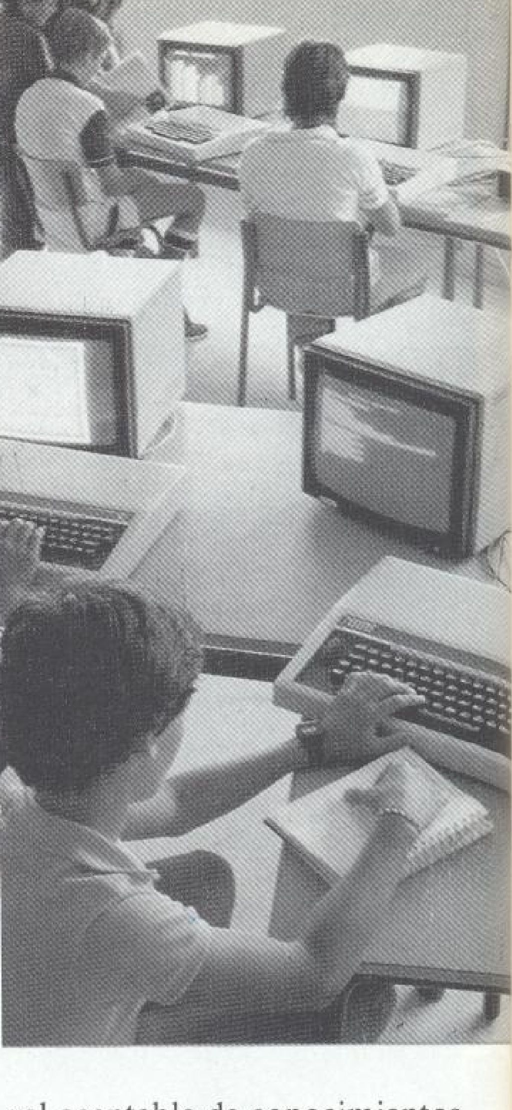
informática a la escuela hasta que no haya, al menos, un ordenador en cada clase al que tengan libre acceso todos los alumnos.

Como veremos, esto es posible y se puede hacer con muy poco dinero. (En el caso del Spectrum, una configuración básica que incluya el ordenador, un grabador de cassettes, un monitor o un televisor y una impresora de 80 columnas con su correspondiente interface, se puede conseguir por poco más de 100.000 pesetas. Por supuesto, los costes pueden reducirse sensiblemente si la impresora es de peor calidad o se comparte entre varias clases.) La verdadera dificultad para realizar este proyecto no es económica, sino ideológica.

Evidentemente, el ordenador así concebido no tiene cabida en un sistema de educación en



Los programas de simulación poseen un alto valor pedagógico.



el que el profesor, subido en una tarima o sentado detrás de una mesa frente a unos pupitres regularmente ordenados, es la única fuente de información. El ordenador únicamente será una herramienta útil en un ambiente en el que predomina la investigación sobre la mera exposición de contenidos, en unas clases en las que es frecuente acudir a múltiples fuentes de información, en sustitución del libro de texto y los apuntes como únicos elementos de referencia.

No tiene ningún sentido ni utilidad el sentar ocasionalmente a un grupo de alumnos frente a una pantalla para que presencien una demostración informática, del mismo modo que podían estar presenciando una proyección de diapositivas o una experiencia de cátedra. Los alumnos no pueden acceder a la informática como simples espectadores: es necesario que utilicen el ordenador y que lo utilicen de un modo habitual, al igual que diariamente hacen uso de sus libros o sus bolígrafos.

El ordenador únicamente será una herramienta útil en un ambiente en el que predomine la investigación.

Con estas afirmaciones no se excluye la utilización del ordenador como un importante elemento de apoyo para el desarrollo de las clases. Simplemente se mantiene que, una vez que el ordenador forma parte de un ambiente educativo, ésta es tan sólo una de sus numerosas aplicaciones, pero no la única.

De un modo general, es posible distinguir los siguientes campos de aplicación dentro de una clase:

a) Enseñanza asistida por ordenador (EAO).
b) Simulaciones.
c) Adquisición de técnicas, destrezas y conceptos generales.
d) Utilización de los paquetes de aplicaciones.

a) Enseñanza asistida por ordenador

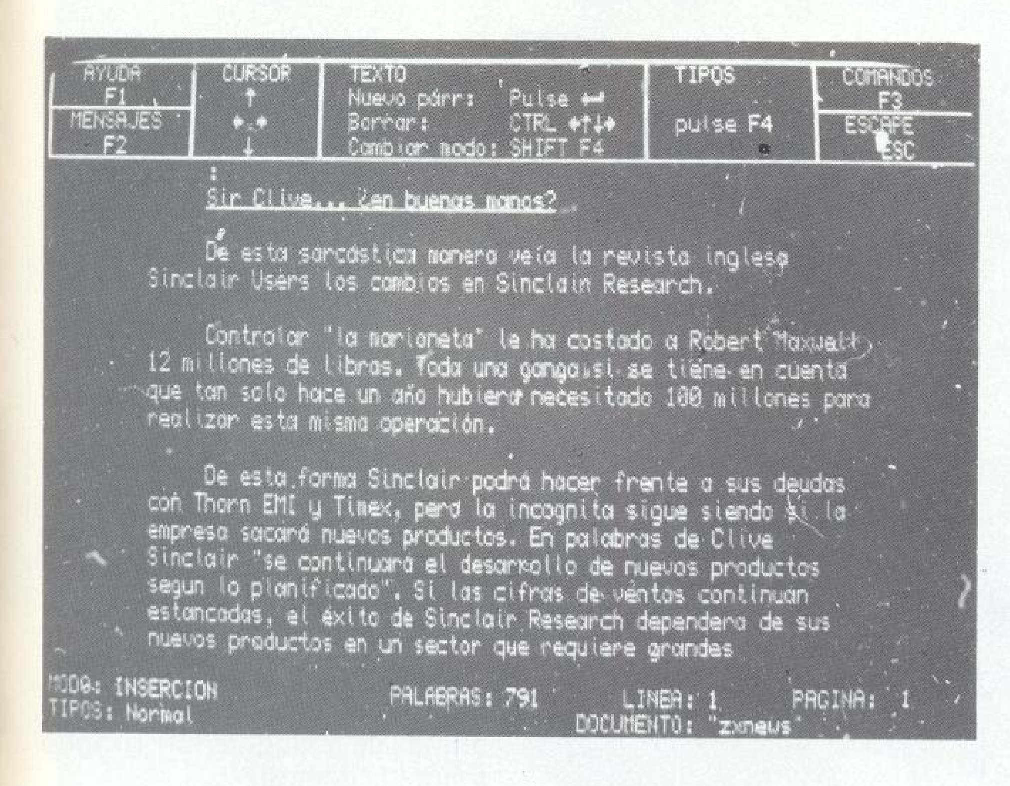
Como tal se entiende la enseñanza que se imparte por medio de un ordenador, que actúa como intermediario entre un profesor que ha diseñado un programa y un alumno que aprende siguiendo las indicaciones de dicho programa.

La mayor o menor utilización de este tipo de recurso dependerá en la mayoría de los casos de la disponibilidad de programas de EAO que existan en el mercado, ya que normalmente el profesor no dispondrá del tiempo o los conocimientos suficientes para elaborar dichos programas.

Desgraciadamente los programas que existen en el mercado, tanto para el Spectrum como para otros ordenadores domésticos son escasos y de baja calidad. Esto se debe, en gran medida, a que están diseñados por personas que tienen escasos conocimientos de pedagogía. Cualquier programa de EAO que pretenda tener una cierta calidad debería estar realizado bien por un equipo de pedagogos y programadores o bien por pedagogos con un nivel aceptable de conocimientos sobre algoritmos y programación.

No obstante, existe una tercera posibilidad: la utilización de los llamados lenguajes de autor. Dichos lenguajes están diseñados para ser utilizados por usuarios no necesariamente versados en programación, pero con buenos conocimientos sobre enseñanza programada.

Con los lenguajes de autor es posible elaborar lecciones interactivas que incluyen textos informativos, incorporan gráficos y sonido, formulan preguntas que esperan una letra, palabra o frase como respuesta y son más o menos versátiles en función de la calidad del lenguaje y de la capacidad del ordenador (así, algunos lenguajes de autor permiten aceptar respuestas que difieran en una letra con respecto a la respuesta prevista; o bien son capaces de almacenar en memoria un número indeterminado de lecciones, o



permiten la definición de varios niveles de dificultad, etcétera).

Hay que señalar, sin embargo, que, por el momento, la capacidad del ordenador es un factor limitante para poder trabajar con este tipo de lenguajes. Así, en el caso de los ordenadores domésticos, no existen en el mercado lenguajes de autor, si bien es posible conseguir algunos programas que imitan de un modo bastante pobre algunas de las facilidades de estos lenguajes.

En contraste con los programas clásicos de EAO, las simulaciones permiten al estudiante la manipulación de variables. Su valor educativo es muy superior.

b) Simulaciones

Este tipo de programas intenta imitar la evolución de una serie de fenómenos cuyo curso se puede manipular. Así, por ejemplo, el aterrizaje de un avión o cualquier otra situación de la vida real.

En contraste con los programas clásicos de EAO, meramente informativos y de instrucción, estos programas permiten al estudiante que descubra la solución a un determinado problema mediante la manipulación de una serie de variables. Son, por tanto, de un valor educativo mucho mayor.

En este sentido, existen programas que simulan experimentos de laboratorio cuyos costes o materiales especializados impiden su realización en una escuela normal (por ejemplo, experimentos de genética, ecología o física nuclear). Asimismo, se puede disponer de juegos que requieren de una determinada estrategia para la resolución de un problema, de simuladores de empresas, de programas que plantean problemas de tipo humanístico que no tienen una única solución, etcétera.

En este campo, al igual que en todos, la sofisticación de los programas depende del tipo de ordenador con el que se trabaje. Sin embargo, en el Spectrum la disponibilidad y la calidad de los materiales es mucho mayor que en el caso de la EAO. Existen en el mercado buenos programas de simulación y además, y esto es importante, un fondo de programas de simulación elaborados por profesionales de la enseñanza. Dicho fondo no está concentrado en una institución o centro determinado, pero es posible disponer de catálogos elaborados por instituciones que trabajan sobre estos temas, tales como el presentado por el comité organizador de las II Jornadas Nacionales sobre Informática en la Enseñanza que se llevaron a cabo en Barbastro, en julio de 1985.

c) Adquisición de técnicas, destrezas y conceptos generales

Existe un conjunto de conocimientos generales que no son el objeto de estudio de ninguna asignatura en concreto, pero que sin embargo posibilitan un

mejor acceso a cualquier tipo de aprendizaje. Son las llamadas metatécnicas que incluyen, entre otras, todas las formas de percepción (visual, auditiva, etc.), el razonamiento lógico-matemático en su sentido más amplio, todas las formas de memoria, la lectura, la escritura, el dibujo, las habilidades necesarias para la realización de cualquier tipo de tarea que requiera de una elevada dosis de concentración y precisión, las habilidades necesarias para la resolución de situaciones que requieran de una respuesta rápida y precisa, etcétera.



Para el aprendizaje y desarrollo de muchas de estas técnicas el ordenador supone una ayuda inestimable. Baste citar, por ejemplo, los múltiples juegos donde la rapidez de reflejos es la principal clave del éxito, o el enorme potencial educativo del lenguaje Logo, tanto en su conocido aspecto geométrico como en otros aspectos menos difundidos, como el tratamiento de listas o la simulación de fenómenos físicos; todos aquellos programas que facilitan la composición musical con escasos o inexistentes conocimientos de solfeo, o aquellos que permiten la realización de diseños y dibujos. En este campo es en el que el Spectrum dispone de una mayor cantidad de programas de calidad, incluido el inestimable lenguaje Logo.

d) Utilización de los paquetes de aplicaciones

Aunque anteriormente ya se hizo referencia a este punto, es interesante señalar las prestaciones que estos paquetes de aplicaciones proporcionan en una clase.

Los alumnos pueden elaborar sus propias bases de datos y utilizarlas posteriormente como fuente de consulta. Así, por ejemplo, la realización de una base de datos de Geografía no encierra ninguna dificultad si se dispone de un programa adecuado.

Las hojas de cálculo permiten a los alumnos el trabajo con grandes masas de datos numéricos, eliminando la pérdida de tiempo que supone el trabajo con interminables listas de números, particularmente en estudios estadísticos o económicos.

Los procesadores de textos ofrecen la posibilidad de mejorar sensiblemente la calidad de los trabajos escritos, e incluso, si el procesador es lo suficientemente bueno, la posibilidad de mejorar su ortografía. Asimismo, ofrecerán a la clase la posibilidd de elaborar un amplio fondo de documentos.

Conclusión

Los ordenadores pueden convertirse en una herramienta de inestimable ayuda para los profesores. Con unos costes relativamente bajos, es posible acceder a un fondo importante de recursos, válido para las actividades más diversas. Para ello no es necesario disponer de conocimientos especializados: basta con utilizar los programas disponibles en el mercado. Muchos de ellos son de fácil manejo y están diseñados para que puedan ser utilizados por usuarios que desconocen los lenguajes de programación.

El Spectrum, dado su bajo coste, su gran difusión entre los alumnos y la gran cantidad de programas existentes, es una máquina a tener en cuenta dentro de la clase, del mismo modo que los libros o la pizarra.

Enrique Sánchez Ludeña

Rutinas de la ROM del ZX Spectrum

La ROM del SPECTRUM contiene multitud de rutinas que pueden ser aprovechadas para optimizar el trabajo de escritura de un programa. Muchas de estas rutinas sólo pueden ser utilizadas desde programas en código máquina, ya que es preciso sumistrarles algunos parámetros para su correcto funcionamiento. Otras, en cambio, pueden ser llamadas tanto por un programa en BASIC como por uno en código máquina.

La forma más usual de llamar a estas rutinas desde el BASIC es: RANDOMIZE USR xxxx donde xxxx es una dirección de la ROM. En los programas que utilicen el generador de números pseudoaleatorios (función RND) no debería emplearse RANDOMIZE, ya que al retornar el control al BASIC, el valor contenido en el par de registros BC es puesto en la variable del sistema SEED (23670/1), utilizada para contener la semilla del siguiente número aleatorio a generar, lo que da lugar a una repetición de la secuencia.

Vamos a comentar algunas de estas rutinas que pueden ser utilizadas desde los programas BASIC:

La primera rutina que nos encontramos en la memoria es lógicamente la que comienza en la dirección 0. A ella accede el procesador cuando se conecta el ordenador o se pulsa el botón de reset. La función de esta rutina es la de inicializar completamente el ordenador y por lo tanto destruir cualquier programa que se encuentre en la memoria. Normalmente no se utiliza dentro de un programa, a no ser para finalizarlo. Exis-

ten otros muchos puntos de la ROM que pueden ser utilizados para provocar un RESET del sistema; algunos de ellos lo hacen de una forma un tanto curiosa, por ejemplo:

RANDOMIZE USR 5480
RANDOMIZE USR 2110
RANDOMIZE USR 8250

La pantalla

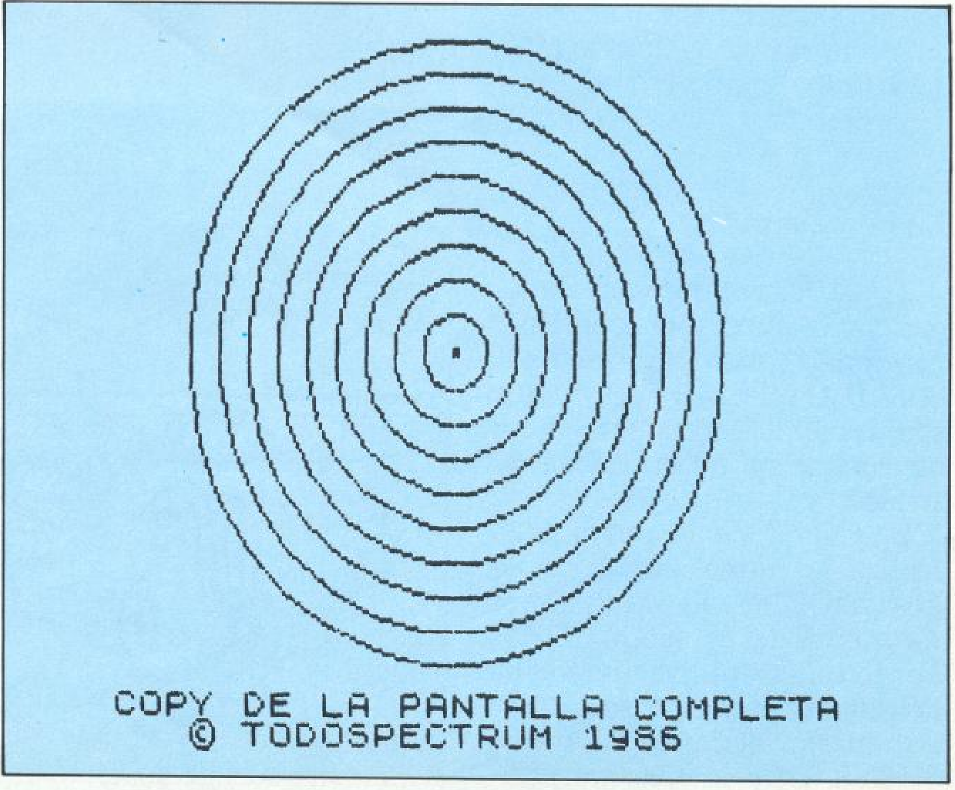
En la dirección 3435 se encuentra la rutina que produce un borrado de pantalla e inicialización de nuevos atributos. La utilización de esta rutina en un programa BASIC no tiene ningún sentido, a menos que se pretenda dificultar su lectura a personas ajenas, puesto que realiza la misma tarea que el comando CLS.

— Utilización de la parte inferior de la pantalla:

Las dos últimas líneas de la pantalla están reservadas para las entradas desde el teclado y mensajes de error del sistema. En ellas no se pueden utilizar los comandos PLOT y DRAW, aunque sí es posible la utilización del comado PRINT con la siguiente sintaxis: PRINT 0, "MENSAJE A IMPRIMIR" o PRINT 0;AT Y, X; "MENSAJE A IMPRIMIR". Estas dos líneas se pueden utilizar en los programas para presentar mensajes de advertencia o ayuda al usuario, y en muchas ocasiones es necesario borrarlos sin afectar al resto de la pantalla. La rutina localizada en la posición 3438 es la encargada de realizar esta función.

En la dirección 3652 se encuentra la rutina que borra solamente una parte de la pantalla; dicha rutina sólo puede utilizarse plenamente desde códi-





go máquina, ya que precisa un parámetro de entrada en el registro B que define el número de líneas a borrar. Una forma más original de borrar la pantalla es utilizar la rutina que comienza en la dirección 3330. La llamada a esta rutina debe ir seguida de un CLS, puesto que deja una línea de la pantalla sin borrar.

— Desplazamiento de pantalla.

En la posición 3582 se encuentra la rutina que produce un desplazamiento (SCROLL) de pantalla una línea hacia arriba. Esta rutina no afecta a la posición actual de impresión. El mismo efecto se cosigue utilizando la llamada a la rutina localizada en la dirección 3190. Un SCROLL similar al de la rutina anterior es el producido por la rutina localizada en la posición 3584, con la diferencia de que esta última sólo desplaza las 14 líneas inferiores. Los programadores de código máquina podrán sacar mucho más partido a esta rutina, ya que se puede definir el número de líneas a desplazar.

La impresora

Todas las rutinas relaccionadas con la impresora solamente funcionan con la impresora ZX o SEIKOSHA GP50, ya que son las únicas diseñadas para conectarse sin necesidad de interface.

El COPY de la pantalla lo realiza la rutina situada en la posición 3756. El resultado es idéntico al del comando COPY, por lo que su utilización queda marginada para el caso indicado anteriormente en la rutina de CLS. Otra interesante posi-

```
   1 GO SUB 100
   5 PRINT #0,AT 0,3;"COPY DE LA
 PANTALLA COMPLETA": PRINT #0,AT
 1,6;"© TODOSPECTRUM 1986"
  10 COPY : LET dir=20672: LET b
af=23296
  20 FOR l=1 TO 2: FOR x=1 TO 8
  30 FOR a=dir TO dir+31: POKE b
af,PEEK a: LET baf=baf+1: NEXT a
  40 LET dir=dir+256: NEXT x
  50 RANDOMIZE USR 3789: REM COP
IAR BUFFER DE IMPRESORA
  60 LET baf=23296: LET dir=2070
4: NEXT l
  70 FOR A=1 TO 21: RANDOMIZE US
R 3190: REM SCROLL
  80 PAUSE 10: NEXT A
  90 RANDOMIZE USR 4766: REM FIN
AL DEL PROGRAMA
 100 FOR A=1 TO 90 STEP 9
 110 CIRCLE 128,88,A: NEXT A
 120 PRINT #0,AT 0,8;"PULSA ESPA
CIO"
 130 RANDOMIZE USR 1415: REM ESP
ERAR ESPACIO
 140 RANDOMIZE USR 3438: REM BOR
RA PANTALLA INFERIOR
 150 PRINT #0,AT 0,0;"ESTE PROGA
MA ES UNA DEMOSTRACION": PRINT #
0,AT 1,2;"DE LA UTILIZACION DE L
A ROM"
 160 RANDOMIZE USR 1415: REM ESP
ERAR ESPACIO
 170 RANDOMIZE USR 3438: REM BOR
RAR PARTE INFERIOR
 180 RETURN
```

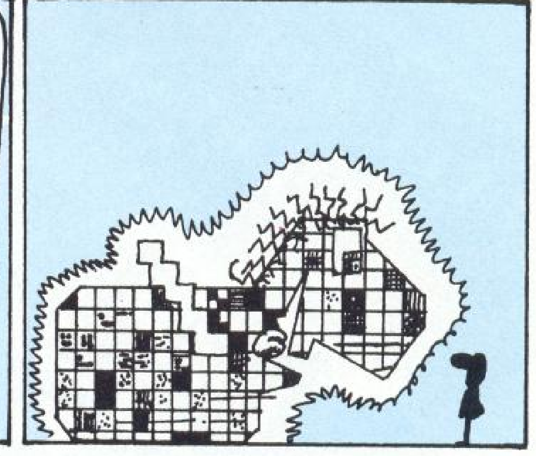


GUSANEZ

por José C. Tomás







bilidad es imprimir el contenido del BUFFER (memoria intermedia de impresora), localizado entre las posiciones 23296 y 23450. Un conjunto de 8 bytes consecutivos forma un caracter de 8 × 8 puntos, en total 32 caracteres que se imprimen como una línea de texto. El listado incluido en este artículo es una aplicación de esta rutina para realizar un COPY de la pantalla completa.

Algunas rutinas curiosas

Comprobación de la tecla SPACE: La rutina localizada en la posición 1415 detiene la ejecución del programa hasta que la barra espaciadora sea pulsada. La llamada a esta rutina puede sustituir a una línea como esta:

90 IF INKEY$ < > "" THEN GOTO 90

La instrucción PAUSE 0 también se puede conseguir mediante una llamada a la ROM; concretamente a la dirección 5598, aunque su utilidad es prácticamente nula.

Algunos BASIC disponen del comando END para finalizar la ejecución de programa. En el Spectrun puede simularse mediante RANDOMIZE USR 4447 o RANDOMIZE USR 4766; la diferencia entre estas dos llamadas radica en que la segunda produce un listado automático del programa, además de finalizar su ejecución.

Si ejecutamos RANDOMIZE USR 5080 en modo directo, observamos que se incorpora una línea 0 con dicha llamada. En cambio, si tecleamos RANDOMIZE USR 5080 :REM (C) TODOSPECTRUM aparecerá la línea 0 con la instrucción REM (C) TODOSPECTRUM.

Otros efectos

RANDOMIZE USR 4700 restaura los colores del BORDER PAPER e INK, borra la pantalla e imprime el mensaje (c) 1982 Sinclair research ltd. El programa existente en memoria no resulta alterado.

PRINT AT Y, X;: RANDOMIZE USR 4755: el efecto conseguido es la impresión del mensaje de Sinclair en cualquier posición de la pantalla.

RANDOMIZE USR 4640: Este es un punto de entrada de la rutina de NEW. El efecto es el mismo, aunque de una forma más suave.

Otro efecto curioso se consigue haciendo un RANDOMIZE USR 3034 después de ejecutar la instrucción POKE USR 3030,100. Pero mejor dejamos que el lector experimente con esta y con todas las direcciones que a continuación se listan:

RANDOMIZE USR 8520
RANDOMIZE USR 1300
RANDOMIZE USR 4795
RANDOMIZE USR 6830
RANDOMIZE USR 7110
RANDOMIZE USR 3180
RANDOMIZE USR 1560
RANDOMIZE USR 5120
RANDOMIZE USR 9000
RANDOMIZE USR 2070
RANDOMIZE USR 2140
RANDOMIZE USR 2304

Para terminar listamos dos programas de demostración que utilizan algunas de estas llamadas.

FELICES CUELGUES.

Octavio López



En el capítulo anterior vimos la forma de saltar de una parte a otra del programa y cómo utilizar subrutinas. Cómo existen saltos absolutos (JP, que ocupan tres bytes y en los que especificamos la dirección absoluta en que debe continuar la ejecución) y saltos relativos (JR, que ocupan dos bytes y en los que lo que marcamos es el desplazamiento a efectuar desde la posición actual del puntero de programa). También vimos la manera de utilizar la instrucción CALL (que llama a una subrutina de la que se regresa con RET), y cómo estas cuatro instrucciones podían ser condicionales, es decir, que se ejecuten o no dependiendo de que se cumpla una determinada condición.

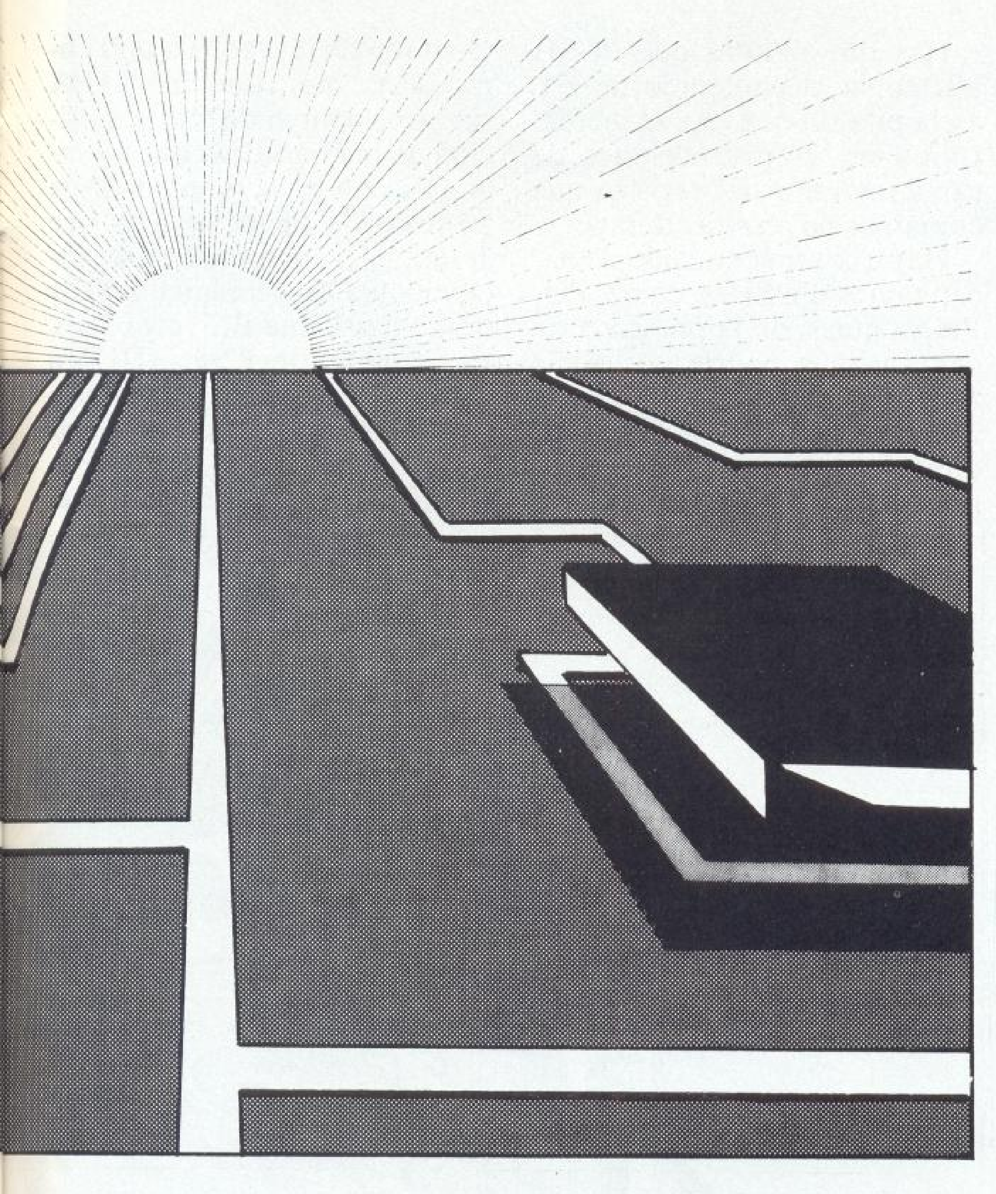
Estas condiciones pueden ser referentes a todas las banderas en el caso de JP, CALL y RET, o sólo a las de acarreo y cero en los saltos relativos. Esto nos permite dar a los programas una potencia mayor que si tuviéramos que contentarnos con seguir un flujo constante si el uso de bucles, bifurcaciones o subrutinas.

Dos instrucciones en una

Supongamos que necesitamos utilizar un bucle que repita un trozo de programa un número determinado de veces y en el que podamos llevar la cuenta de las veces que lo hemos repetido (algo así como un FOR-NEXT del BASIC). Bastará dar a un registro un determinado valor, ejecutar el trozo de programa, decrementar el registro y, si el resultado no es cero, volver a ejecutar el bucle. Esto es algo que se realiza tan a menudo que se creó una instrucción que hiciera las cosas más cómodas. Esta es DJNZ, y lo que hace es decrementar el registro B y saltar si el resultado no es cero o una distancia (entre –128 y 127 bytes) de la posición de memoria que sigue a esta instrucción.

Vamos a intentar crear un bucle que imprima 100 caracteres en pantalla y regrese al BASIC. Para ello podemos dar a B el valor 100, a A 32, sumar ambos y, tras imprimir el resultado (que queda en A) por medio del ya utilizado RST 16, hacer DJNZ. De este modo se consigue que el código a imprimir no sea menor de 32, es decir, los caracteres de control, lo que, impresos sin su correspondiente orden, podrían despistar a la rutina de impresión.

Antes de utilizar el RST 16 abriremos el canal 2, de modo que la impresión se realice en la parte principal de la pantalla. Esto es necesario porque el sistema operativo del BASIC, que hasta ese momento estaba vigente, suele utilizar el canal 1 (parte baja de la pantalla) para imprimir los comandos directos que tecleemos. Para ello utilizaremos una rutina de la ROM llamada CHAN-OPEN, que comienza en la dirección 5633



(1601h) y a la que habremos de llamar (con CALL) tras haber dado al registro A el número de canal (0 y 1 para la parte baja de la pantalla, 2 para la parte principal y 3 para la impresora). El listado final debe quedar como el de la figura 1.

```
10          ORG   60000
20
30          LD    A,2
40          CALL  5633
50          LD    B,100
60  BUCLE   LD    A,32
70          ADD   A,B
80          RST   16
90          DJNZ  BUCLE
100         RET
```

Figura 1

Intercambio entre registros

Existen seis instrucciones que permiten permutar unos registros por otros o por el contenido de la cola del *stack,* todas ellas comienzan por EX (de EXchange, cambio).

EX DE,HL introduce el contenido de DE en HL y el de HL en DE. EX (SP), HL hace lo mismo pero utilizando en lugar de DE el contenido de la posición de memoria a la que apunta el puntero de pila, o lo que es lo mismo, el último valor introducido en la pila o *stack* (más adelante explicaremos lo que es). Esta operación también puede realizarse usando IX o IY en lugar de HL.

En el capítulo segundo vimos que existía un grupo de registros alternativo que, aunque no puedan utilizarse en la forma normal, pueden intercambiarse con los del grupo principal, y ser usados mientras estos últimos quedan en «el banquillo». EX AF, AF' lo hace con el par AF, lo que nos permite salvaguardar el contenido del acumulador mientras lo utilizamos para otra cosa, o hacer lo propio con los *flags* o banderas, almancenando los resultados (en cuanto a acarreo, paridad, etc.) de una operación de la que va a depender una decisión posterior. EXX a secas cambia de golpe BC, HL y DE por BC', HL' y DE'. Todas éstas son instrucciones potentes, pero bastante delicadas, por lo que hay que utilizarlas cuidadosamente.

```
10     ORG   60000
20
30     LD    HL,16384
40     LD    DE,50000
50     LD    BC,6912
60     LDIR
70     RET
```

Figura 2

El stack o pila de máquina

El *stack* comúnmente se suele comparar con una pila o montón de platos, por lo que imaginémonos que trabajamos en un restaurante y tenemos ante nosotros una buena cantidad de ellos. Cada plato puede contener una dato (por ejemplo, una tortilla francesa) y, en condiciones normales, sólo podemos acceder al que se encuentra en la cima del montón, es decir, el último que hayamos

puesto. Si intentáramos meter o sacar un plato de los del centro de la pila, nos arriesgaríamos a que toda ella se viniera abajo, con el consiguiente despido por parte del dueño del restaurante.

En realidad, la pila consiste en una zona de memoria cuyo final queda marcado por el par de registros SP (Stack Pointer o puntero de pila). En ella se almacenan las direcciones de retorno de cada CALL que se ejecute, que serán recuperadas por los correspondientes RET. La pila crece hacia abajo en la memoria, por lo que cada vez que introduzcamos un dato en ella se decrementará en dos *bytes* SP y se almacenará el dato en la dirección a la que quede señalado éste. Esto lo podemos hacer por medio de la instrucción PUSH seguida de AF, BC, HL, DE, IX o IY. Para tomar un dato de la pila usaremos POP, que almacena los dos *bytes* a los que se apunta SP en el par de registros que especificamos e incrementa SP en dos *bytes.*

Otras instrucciones que afectan a SP son las que nos permiten cargar en él el contenido de HL, IX o IY mediante LD SP, HL o similares. Esto cambia de sitio la «cola» del *stack,* lo que, utilizado con cuidado, puede resultar muy interesante. El comando CLEAR del BASIC seguido de un número, cambia de sitio la pila situando su base en la dirección que especifiquemos.

El *stack* puede ser utilizado de diversas formas: para salvar el contenido de un par de registros mientras se ejecuta una subrutina que los modifica o también para cargar en un par de registros el contenido de otro par. Por ejemplo, LD BC, HL no existe en el juego de instrucciones del Z-80, pero podemos emularla haciendo:

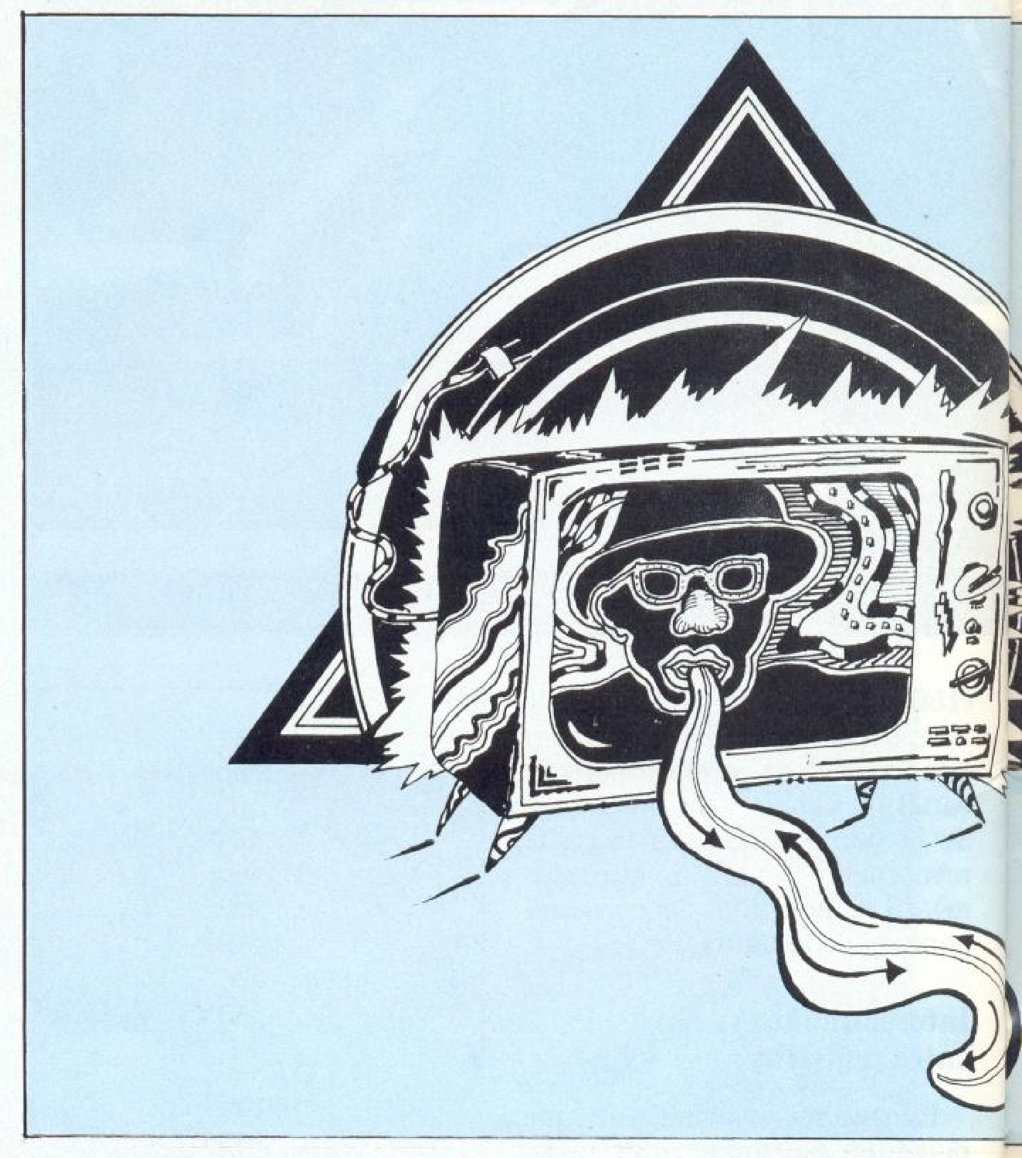
```
PUSH HL
POP BC
```

Hay que resaltar que PUSH HL copia el contenido de HL en la pila sin que éste se pierda. Como en la instrucciones de carga, el valor del registro que copiamos no resulta alterado.

Hay que tener cuidado de hacer tantos PUSHes como POPes ya que si no la pila crecería demasiado y podría invadir la zona de RAM utilizada por el sistema operativo. Además las direcciones de retorno quedarían sepultadas bajo esos datos, y la vuelta se haría al azar, con resultados poco agradables.

Transferencia de bloques de memoria

Existen dos instrucciones que permiten mover un bloque de memoria de una zona a otra cómoda y rápidamente. LDIR carga el contenido de la posición de memoria a la que apunta DE con el contenido de la posición de memoria a la que apunta HL, incrementa HL, incrementa DE, decrementa BC y repite hasta que BC valga cero. Es decir, copia de (HL) a (HL+BC) a partir de (DE). LDDR es similar, pero en lugar de incrementar HL y DE los decrementa, lo cual es útil si el bloque de destino solapa al de origen y está por encima en la memoria. LDI y LDD hacen lo mismo, pero sin repetir hasta que BC sea cero. Estas instrucciones no modifican la bandera Z, pero podemos saber cuándo se ha llegado a este punto tes-



teando la bandera P/V, que estará desactivada sólo en ese caso.

Sabiendo esto podemos crear una rutina que copie el trozo de memoria que corresponde al archivo de presentación visual en la parte alta de la memoria, de donde podremos regresarla a la pantalla cuando queramos. Sólo nos hace falta saber que el archivo de pantalla, incluido el de atributos, comienza en la dirección 16384 y ocupa 6912 *bytes*. Las dos cortas rutinas de las figuras 2 y 3 harán el milagro.

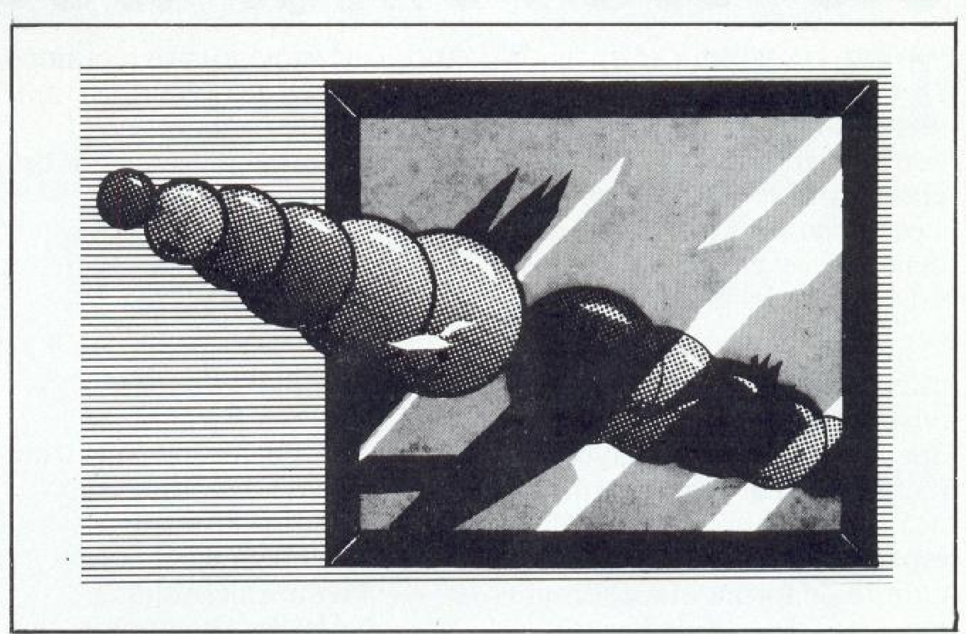
Otra aplicación de estas instrucciones es llenar un bloque con un *byte* determinado. Puede ser útil para borrar una zona llenándola de ceros (p. e., la pantalla) o cualquier otro valor. El sistema es sencillo: dar a HL y DE dos valores consecutivos y cargar en la posición marcada por HL el *byte* que nos interesa. LDIR copiará esa posición en la siguiente, ésta en la tercera, y así tantas veces como indique el par BC.

En la figura 4 aparece el listado de una subrutina que llena el archivo de atributos con el número 58. Esto corresponde a papel blanco y tinta roja, por lo que toda la pantalla cambiará de color. Hay que tener en cuenta que no son atributos permanentes, ya que para eso tendríamos que modificar la correspondiente variable del sistema.

Cómo localizar un número en la memoria

Hay cuatro instrucciones que se encargan de las comparaciones: CPIR compara el byte a que apunta el par HL con el contenido del acumulador y, si no son iguales, incrementa HL y repite hasta que BC valga cero. Al igual que con LDIR, existe la posibilidad de que HL sea decrementado (CPDR) o de que no repita (CPI y CPD). En todo caso, Z señalará si se ha encontrado el *byte* buscado (si es así HL apuntará a la dirección siguiente a la que nos interesa) y P/V quedará activada si BC es distinto de cero.





```
10        ORG   60000
20
30        LD    HL,50000
40        LD    DE,16384
50        LD    BC,6912
60        LDIR
70        RET
```

Figura 3

```
10        ORG   60000
20
30        LD    HL,16384
35        LD    (HL),58
40        LD    DE,16385
50        LD    BC,6911
60        LDIR
70        RET
```

Figura 4

Programas 60

Gráficas de cinemática

Este programa se encuadra dentro de la filosofía de trabajo del grupo ABAX, constituido por profesores de enseñanza secundaria que elaboran *software* con el que el ordenador es considerado como un auxiliar del profesor.

El Spectrum simula el movimiento de uno o dos móviles (visualizados como cochecitos) que describen una trayectoria rectilínea. De forma simultánea, se construyen las gráficas espacio-tiempo y velocidad tiempo de forma que el programa será de utilidad para los profesores y estudiantes de Física (de modo especial, dentro del nivel de BUP y FP), en el estudio del movimiento rectilíneo uniforme y del rectilíneo uniformemente variado.

El programa presenta dos opciones generales:

- caso de un móvil único
- caso de dos móviles

1. Un móvil

En esta opción, el Spectrum nos solicitará los valores de:

— Velocidad inicial.
— Aceleración.
— Tiempo a estudiar.
— Número de puntos que deseamos obtener.
— Si desea ser el usuario el que avance el coche o bien si queremos que sea el Spectrum quien lo mueva.
— Si deseamos visualizar sobre la trayectoria las diferentes posiciones.

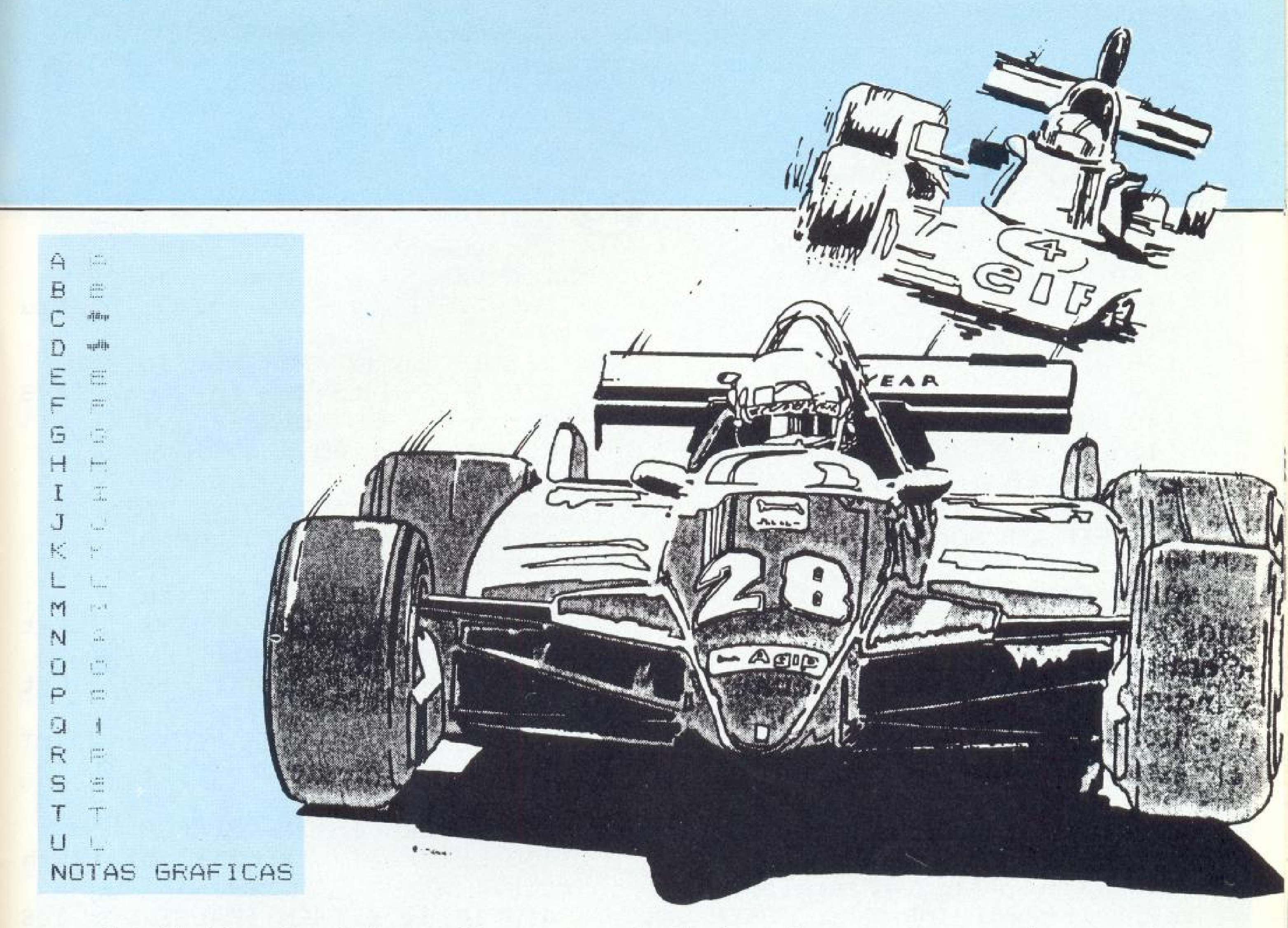
Una vez indicados estos datos, en pantalla se simulará el movimiento del coche a la vez que se construyen las gráficas x-t y v-t.

2. Dos móviles

En la versión del programa que se presenta, esta opción se refiere a la situación de dos móviles (coches) que parten simultáneamente hacia su encuentro.

Los datos que nos solicita el ordenador son:

```
  10 REM **C I N E M A T I C A*
*
  15 REMMiguel Gisbert Brianso
-1985
  16
  17
  20 REM Caracteres del usuario
  21 GO SUB 9000
  23
  24
  25 REM **PORTADA**
  26 GO SUB 7000
  28
  29
  30 REM instrucciones
  35 BORDER 1: PAPER 7: INK 2: C
LS
  40 PRINT AT 2,4; BRIGHT 1; FLA
SH 1;"PROGRAMA DE CINEMATICA": P
RINT AT 4,7;"Uno o dos mo";CHR$
8; OVER 1;"'viles";AT 5,4;"Traye
ctoria rectili";CHR$ 8; OVER 1;"
'nea"
  45 PRINT AT 10,0; INK 3;"Colab
.:E.LALANA & S.MANRIQUE": PRINT
AT 9,7; INK 1; BRIGHT 1;"Autor:M
.GISBERT"
  50 INK 0: PRINT : PRINT "  Cat
edra";CHR$ 8; OVER 1;"'ticos de
Bachillerato": PRINT TAB 5;"Miem
bros del grupo"; FLASH 1;"ABAX"
  55 GO SUB 9900
  60 BORDER 5: PAPER 1: INK 7: C
LS
```



— Celeridad de cada móvil.

— Sus aceleraciones.

— Distancia que les separa.

— Si desea ser el usuario el que avance los coches (mediante el teclado) o bien dejamos esta tarea al Spectrum.

El programa simula el movimiento de los dos coches a la vez que construye las gráficas. Después nos mostrará los valores de diversas magnitudes en el instante del encuentro (o choque):

— Tiempo transcurrido.

— Lugar en el que sucede.

— Velocidad de cada coche en el instante del encuentro, valores que, sin duda, son de utilidad para la resolución de problemas.

Miguel Gisbert Briansó

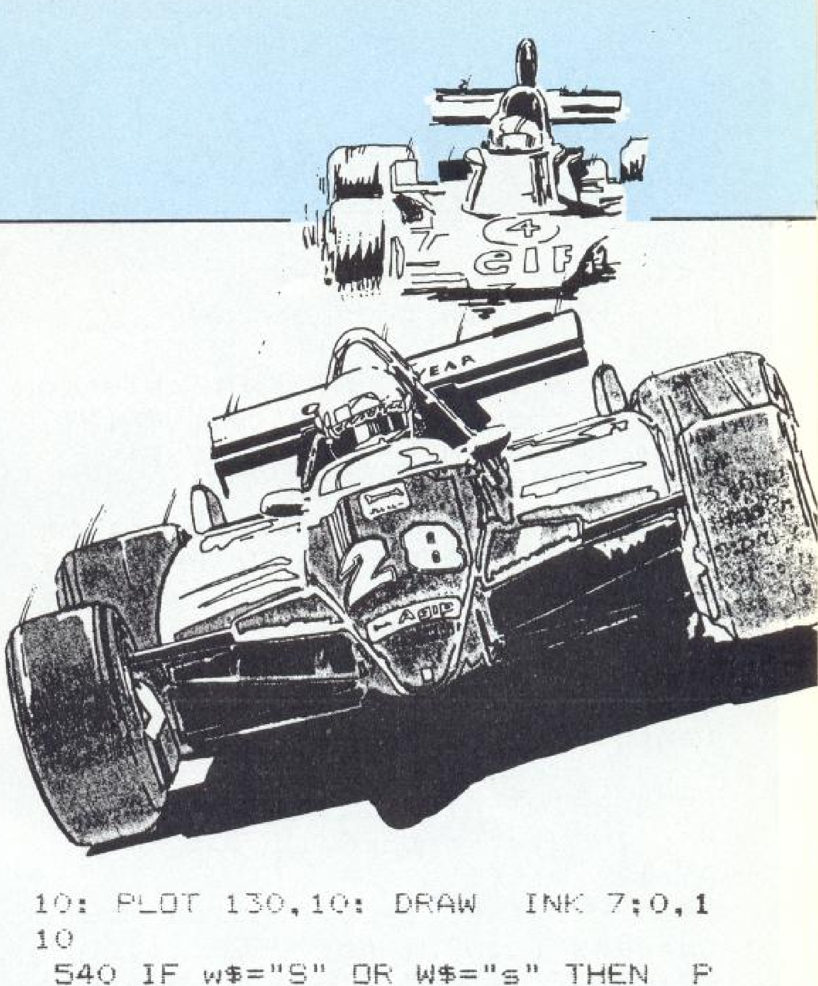
```
  65 PRINT "Este programa se enc
uadra dentrode la filosofia del
grupo ABAX:     ";: PRINT  INK 6;
" el uso del ordenador como
  un auxiliar del profesor"
  70 INK 7: PRINT : PRINT "El or
denador simula el movimien-to de
 uno o dos moviles que des-cribe
n una trayectoria  rectili-nea y
,simultaneamente,construye la gr
afica x/t y la v/t"
  75 PRINT : PRINT "Para el caso
 de dos moviles que parten simul
taneamente hacia su encuentro se
 calcula el tiempo  y lugar de l
a colision,asi como las velocida
des en dicho instan-te."
  80 GO SUB 9900: INK 7
  90 PRINT : PRINT : PRINT "Todo
s los datos se introduciran en u
nidades del sistema SI"
  95 PRINT : PRINT :
 100 PRINT "Respecto a la aceler
acion:se in-dicara negativa si e
s mov.retar-dado"
 105 PRINT : PRINT : PRINT  FLAS
H 1;"Pulse"; FLASH 0;: PRINT " l
a opcion deseada:": PRINT : PRIN
T TAB 6;"1:un movil";TAB 6;"2:do
s moviles"
 110 IF INKEY$="" THEN  GO TO 11
0
 120 IF INKEY$="1" THEN  GO TO 9
00
```

Programas

60

```
 130 IF INKEY$<>"2" THEN  CLS :
GO TO 105
 136
 137
 138
 139
 140 REM caso de dos moviles
 141
 150 BORDER 3: CLS : PRINT AT 10
,0;"celer. movil izq.=": INPUT v
o1: BEEP .1,10: PRINT AT 10,20;v
o1;"m/s"
 151 PRINT AT 12,0;"celer. movil
 dcha.=": INPUT vo2: BEEP .1,5:
PRINT AT 12,20;vo2;"m/s"
 155 LET vo2=-vo2
 170 PRINT AT 14,0;"distancia se
paracion=": INPUT ,d: BEEP .2,15
: PRINT AT 14,23;d;"m"
 171 IF d<=0 THEN  CLS : PRINT A
T 10,3;"Pulso d=";d;" IMPOSIBLE"
: GO SUB 9900: GO TO 150
 180 PRINT AT 16,0;"aceleracion
movil izq.=": INPUT a1: BEEP .1,
4: PRINT AT 16,25;a1;"m/s";"²"
 181 PRINT AT 17,0;"aceleracion
movil dcha.=": INPUT a2: BEEP .1
,18: PRINT AT 17,25;a2;"m/s";"²"
 185 LET a2=-a2
 186 BEEP .5,10: PRINT AT 19,2;
INVERSE 1;"Desea ser Ud. quien a
vance el "; INVERSE 0;TAB 8; INV
ERSE 1;"movil(s/n)?"; INVERSE 0
 187 LET w$=INKEY$
 188 IF w$<>"s" AND w$<>"S" AND
w$<>"n" AND w$<>"N" THEN  GO TO
187
 195 IF a1=a2 THEN  LET tch=d/(v
o1-vo2): GO TO 255
 210 LET raiz=(vo2-vo1)*(vo2-vo1
)+(2*(a1-a2)*d)
 220 IF raiz<=0 THEN  CLS : PRIN
T AT 10,10;"NO CHOCAN": GO SUB 9
900: GO TO 150
 230  LET dis=SQR raiz
 240 LET tp=((vo2-vo1)+dis)/(a1-
a2)
 250 LET tn=((vo2-vo1)-dis)/(a1-
a2)
 255 DEF FN x(u)=vo1*u+.5*a1*u*u
 256 DEF FN v(u)=vo1+a1*u
 258 DEF FN w(u)=vo2+a2*u
 260 DEF FN y(u)=d+vo2*u+.5*a2*u
*u
 265 IF a1=a2 THEN  GO TO 320
 275 IF SGN tp=SGN tn THEN  IF S
GN tp=-1 THEN  BEEP 1,10: PRINT
"NO CHOCAN": GO SUB 9900: GO TO
150
 285 IF SGN tp<>SGN tn THEN  IF
SGN tp=1 THEN  LET tch=tp
 295 IF SGN tp<>SGN tn THEN  IF
SGN tn=1 THEN  LET tch=tn
 305 IF SGN tp=SGN tn THEN  IF t
p>tn THEN  LET tch=tn
 315 IF SGN tp=SGN tn THEN  IF t
p<tn THEN  LET tch=tp
 320 IF tch<=0 THEN  CLS : PRINT
 "NO CHOCA": GO SUB 9900: GO TO
150
 330 BORDER 1: INK 7: CLS
 400 LET xe=(vo1*tch)+(.5*a1*tch
*tch)
 410 IF xe>d THEN  PAUSE 20: CLS
: PRINT AT 10,0;"No choca entre
 los dos moviles": GO SUB 9900:
GO TO 105
 420 REM ahora choca entre ambos
 425 PLOT 0,170: DRAW  BRIGHT 1;
255,0: PRINT AT 0,0;"█ ";AT 0,31;
" █"
 431 PLOT 10,10: DRAW 0,115: PLO
T 10,10: DRAW 105,0
 432 LET n=10: LET e=31/d: PRINT
 AT 7,0;"x";AT 7,15;"v";AT 20,14
;"t"
 433 LET pas=tch/n
 434 IF FN x(tch+2*pas)*e>31 THE
N  LET n=n+1: GO TO 433
 435 LET escala=105/d
 437 PRINT AT 21,5;"
"
 438 IF FN y(tch+2*pas)*e<0 THEN
 LET n=n+1: GO TO 433
 439 PLOT 0,170: DRAW  BRIGHT 1;
255,0: PRINT AT 0,0;"▀▀";AT 0,31;
"▀▀"
 440 GO SUB 6000: PRINT AT 21-(1
0+(-men*ev))/8,15;"0";TAB 30;"t"
: PLOT 130,5: DRAW 0,120: PLOT 1
```

```
30,10+(-men*ev): DRAW 110,0: LET
 cont=0: LET t=0
 441 LET t=t+pas: LET cont=cont+
1
 442 IF t>tch+(2.01*pas) THEN  G
O TO 600
 443 PRINT AT 1,15;"
      "
 445 PRINT AT 0,FN x(t-pas)*e;"
"
 446 PRINT AT 0,FN y(t-pas)*e;"
"
 447 PRINT AT 0,0;"▓ ";AT 0,31;" ▓
": PLOT 0,170: DRAW  BRIGHT 1;25
5,0: IF cont<>n THEN  PRINT AT 0
,FN x(t)*e; INK 7;"▓"
 448 PRINT AT 1,23;"t=";INT (t*1
0+.5)/10;"s": PRINT AT 1,16;"num
.";cont;AT 1,4;"fotos=";n+2
 449 PRINT AT 0,FN y(t)*e; INK 3
;"▓"
 456 IF a2=0 THEN  PLOT 10,115:
DRAW  INK 3;((t*105)/(tch+2*pas)
),-(d-FN y(t))*escala: PLOT 130,
10+(-men*ev)+vo2*ev: DRAW  INK 3
;((t*110)/(tch+2*pas)),0
 457 IF a1=0 THEN  PLOT 10,10: D
RAW ((t*105)/(tch+2*pas)),FN x(t
)*escala: PLOT 130,10+(-men*ev)+
vo1*ev: DRAW ((t*110)/(tch+2*pas
)),0
 458 IF a2<>0 THEN  PLOT 130,10+
(-men*ev)+vo2*ev: DRAW  INK 3;((
t*110)/(tch+2*pas)),FN w(t)*ev-F
N w(0)*ev: FOR r=t-pas TO t STEP
 pas/5: PLOT  INK 3;10+((r*105)/
(tch+2*pas)),10+FN y(r)*escala:
NEXT r
 459 IF a1<>0 THEN  PLOT 130,10+
(-men*ev)+vo1*ev: DRAW ((t*110)/
(tch+2*pas)),FN v(t)*ev-FN v(0)*
ev: FOR m=t-pas TO t STEP pas/5:
 PLOT 10+((m*105)/(tch+2*pas)),1
0+FN x(m)*escala: NEXT m
 500 BEEP .1,20
 510 IF cont=n THEN  PRINT AT 0,
FN y(t)*e; INK 4;"▓▓": FOR b=10 T
O 20 STEP 2: BEEP .2,b: NEXT b
 520 PLOT 10,10: DRAW  INK 7;0,1
```

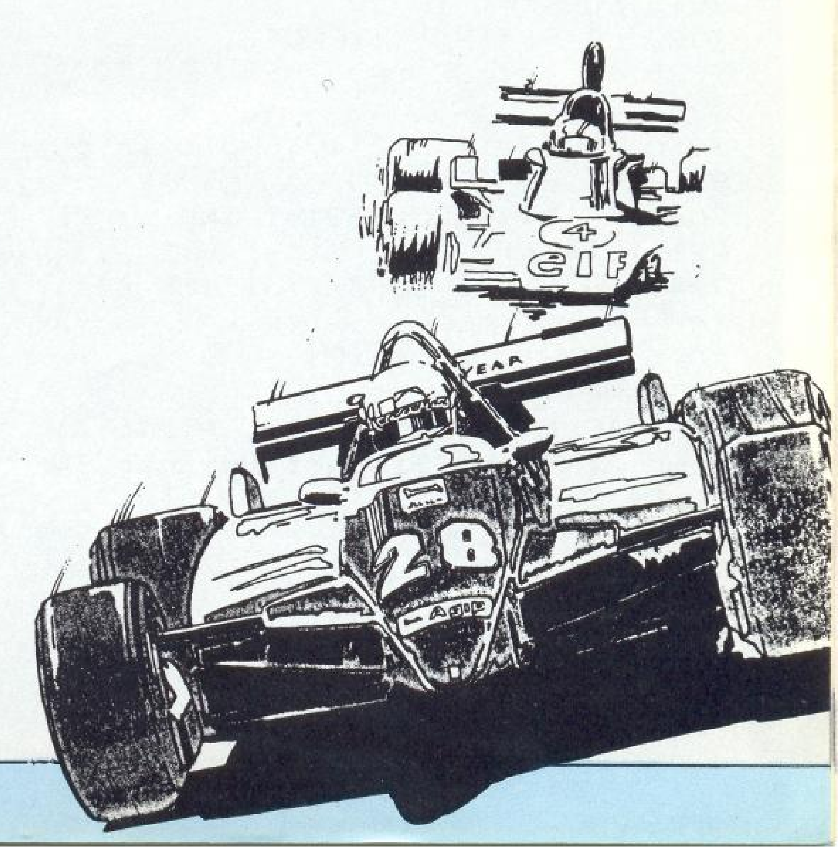


```
10: PLOT 130,10: DRAW  INK 7;0,1
10
 540 IF w$="S" OR W$="s" THEN  P
RINT AT 21,1;"Pulse una tecla pa
ra seguir": PAUSE 0: PRINT AT 21
,1;"                              "
: GO TO 441
 550 IF w$="n" OR w$="N" THEN  G
O TO 441
 600 PRINT AT 1,1;"
                     "
 620 PRINT AT 21,2; FLASH 1;"Pul
se una letra para seguir"
 630 PAUSE 0
 640 BORDER 3: CLS
 650 PRINT AT 0,10;"RESULTADOS":
 PRINT AT 0,10; OVER 1;"________
__"
 660 PRINT AT 3,3;"Punto de encu
entro=";INT (FN x(tch));"m"
 670 PRINT AT 5,3;"Tiempo=";INT
(tch*10+.5)/10;"s"
 675 PRINT : PRINT : PRINT "velo
cidad en la colision:": PRINT :
PRINT TAB 6;"izq.= ";INT (FN v(t
ch)*10+.5)/10;"m/s": PRINT TAB 5
;"dcha.=";INT (FN w(tch)*10+.5)/
10;"m/s"
 700 GO SUB 9900
 720 GO TO 105
 895
 898
```

Programas


```
 899
 900 REM caso deun coche
 905
 910 CLS : PRINT AT 6,0;"celerid
ad del coche=": INPUT vo: PRINT
AT 6,20;vo;"m/s": BEEP .2,8
 915 LET vo=ABS (vo)
 920 PRINT AT 10,6;"aceleracion=
": INPUT a: BEEP .2,12: PRINT AT
 10,20;a;"m/s";"2"
 921 IF vo=0 AND a=0 THEN  CLS :
 PRINT AT 10,0;"Con vel inic=0 y
 sin aceleracion,!!!no hay movim
iento!!!": GO SUB 9900: GO TO 90
0
 924 PRINT AT 12,5;"tiempo final
=": INPUT tf: BEEP .2,30: PRINT
AT 12,20;tf;"s"
 925 PRINT AT 14,0;"indicar num.
 pausas": INPUT np: BEEP .1,20:
PRINT AT 14,0;"       num de pasus
as: ";np
 930 LET rtve=0
 933 PRINT AT 16,2;"Mueve Ud. el
 coche(S/N)?": BEEP .3,5
 934 LET w$=INKEY$: IF W$="" THE
N  GO TO 934
 935 IF W$<>"s" AND w$<>"S" AND
w$<>"n" AND w$<>"N" THEN  GO TO
933
 936 PAUSE 2: BEEP .2,10: PRINT
AT 18,2;"Pulse S si desea ver to
das las posiciones": PAUSE 2
 937 LET u$=INKEY$: IF u$="" THE
N  GO TO 937
 938 IF u$="s" THEN  LET u$="S"
 939 BORDER 1: PAPER 1: CLS
 940 DEF FN r(u)=vo*u+.5*a*u*u
 942 DEF FN s(u)=ABS (vo+a*u)
 950 LET in=10: IF a>=0 THEN  LE
T tin=0: LET din=0
 952 LET df=FN r(tf)
 953 LET escala=105/df: IF a<0 T
HEN  LET tin=(-vo/a): LET din=FN
 r(tin): GO SUB 9800
 955 PRINT AT 7,0;"x";AT 6,13;"!
v!": PRINT AT 20,30;"t": PRINT A
T (22-(in+10)/8),14;"t"
 960 PLOT 10,10: DRAW 0,115: PLO
T 10,in: DRAW 105,0: PLOT 130,10
: DRAW 0,115: PLOT 130,10: DRAW
110,0
 965 PLOT 0,170: DRAW  BRIGHT 1;
255,0: PRINT AT 0,0;"[illegible]"
 970 IF a>=0 THEN  LET ev=110/FN
 s(tf)
 971 IF a<0 THEN  LET ev=110/vo:
 IF FN s(tf)>vo THEN  LET ev=110
/FN s(tf)
 975 LET e=31/df: IF a<0 THEN  L
ET e=31/din: IF df<0 THEN  LET e
=31/(din-df)
 976 LET eux=255/df: IF a<0 THEN
  LET eux=255/din: IF df<0 THEN
 LET eux=255/(din-df)
 980 PRINT AT 21,5;"
                       "
 985 LET cont=0
 990 PRINT AT 1,4;"fotos=";np
1000 PRINT AT 4,0; INK 5;"vo=";v
o;"m/s";TAB 12;"a=";a;"m/s2";TAB
 24;"tf=";tf;"s": INK 7
1200 FOR t=0 TO tf+.0001 STEP tf
/np
1205 LET cont=cont+1
1210 PRINT AT 1,16;"
                 "
1220 IF t<>0 THEN  IF df>=0 THEN
  PRINT AT 0,FN r(t-tf/np)*e;"  "
1221 IF t<>0 THEN  IF df<0 THEN
 PRINT AT 0,FN r(t-tf/np)*e-FN r
(tf)*e;"  "
1222 IF t<>0 THEN  IF df>=0 THEN
  IF u$="S" THEN  PLOT 0,170: DR
AW FN r(t-tf/np)*eux,0: DRAW 0,4
: DRAW 0,-6
1223 IF t<>0 THEN  IF df<0 THEN
 IF u$="S" THEN  PLOT 0,170: DRA
W FN r(t-tf/np)*eux-FN r(tf)*eux
,0: DRAW 0,4: DRAW 0,-6
1224 IF df>=0 THEN  PRINT AT 0,F
N r(t)*e;"[illegible]"
1225 IF df<0 THEN  PRINT AT 0,FN
 r(t)*e-FN r(tf)*e;"[illegible]"
1226 PLOT 0,170: DRAW  BRIGHT 1;
255,0: PRINT AT 0,0;"[illegible]": IF a<0
THEN  IF df<0 THEN  PRINT AT 0,-
FN r(tf)*e;"[illegible]"
```

```
1227 BEEP .1,20
1229 IF t=0 THEN  PRINT AT 1,23;
"t=0 s": IF vo<>0 THEN  PLOT 126
,10+FN s(0)*ev: DRAW 8,0
1230 IF t<>0 THEN  PRINT AT 1,16
;"num.";cont-1;AT 1,23;"t=";INT
(t*10+.5)/10;"s"
1233 IF a<0 THEN  IF t>tin THEN
 IF INT (FN s(t)*10+.5)/10>=1 TH
EN  PRINT AT 2,5;"v=-";INT (FN s
(t)*10+.5)/10;"m/s";"   x=";INT
(FN r(t)+.5);"m": GO TO 1250
1234 IF a<0 THEN  IF t>tin THEN
 IF INT (FN s(t)*10+.5)/10<1 THE
N  PRINT AT 2,5;"v=";INT (FN s(t
)*10+.5)/10;"m/s";"   x=";INT (F
N r(t));"m": GO TO 1250
1235 PRINT AT 2,5;"v=";INT (FN s
(t)*10+.5)/10;"m/s";"   x=";INT
(FN r(t)+.5);"m"
1250 IF a=0 THEN  PLOT 10,in: DR
AW ((t*105)/(tf)),FN r(t)*escala
: PLOT 130,10+FN s(t)*ev: DRAW (
(t*110)/(tf)),0
1259 IF a>0 THEN  IF t<>0 THEN
PLOT 130,10+FN s(0)*ev: DRAW ((t
*110)/(tf)),FN s(t)*ev-FN s(0)*e
v: FOR m=t-tf/np TO t STEP tf/np
/5: PLOT 10+((m*105)/(tf)),in+FN
 r(m)*escala: NEXT m
1260 IF a<0 THEN  IF t<>0 THEN
IF t<=tin+.001 THEN  PLOT 130,10
+FN s(0)*ev: DRAW ((t*110)/(tf))
,FN s(t)*ev-FN s(0)*ev: FOR m=t-
tf/np TO t STEP tf/np/5: PLOT 10
+((m*110)/(tf)),in+FN r(m)*escal
a: NEXT m: GO TO 1300
1261 IF rtve=0 THEN  IF a<0 THEN
  IF t<>0 THEN  IF t>tin THEN  L
ET rtve=1: PLOT 130,10+FN s(0)*e
v: DRAW ((tin*110)/(tf)),FN s(ti
n)*ev-FN s(0)*ev: FOR m=tin-tf/n
p TO tin STEP tf/np/5: PLOT 10+(
(m*110)/(tf)),in+FN r(m)*escala:
 NEXT m
1265 IF a<0 THEN  IF t<>0 THEN
IF t>tin THEN  PLOT 130+(tin*110
/tf),10: DRAW (110*(t-tin)/tf),F
N s(t)*ev: FOR m=t-tf/np TO t ST
EP tf/np/5: PLOT 10+((m*110)/(tf
)),in+FN r(m)*escala: NEXT m
1300 BEEP .1,50
1320 IF w$="s" OR w$="S" THEN  P
RINT AT 21,2;"Pulse una tecla pa
ra seguir": PAUSE 0: PRINT AT 21
,1;"
  "
1400 NEXT t
1500 PRINT AT 21,2; FLASH 1;"Pul
se una letra para seguir"
1520 PAUSE 0
1530 CLS
1550 PAUSE 10: GO TO 105
5996
5997
5998
5999
6000 REM subrutina de velocidad
6001
6005 LET tj=tch+2*pas
6010 IF a1<0 THEN  GO TO 6500
6020 IF FN v(tj)>=FN w(tj) THEN
 LET may=FN v(tj)
6030 IF FN v(tj)<FN w(tj) THEN
LET may=FN w(tj)
6040 IF a2<=0 THEN  LET men=FN w
(tj)
6050 IF a2>0 THEN  LET men=vo2
```



Programas

60

```
6100 LET ev=110/(may-men): RETUR
N
6500 IF vo1>=FN w(tj) THEN  LET
may=vo1
6510 IF vo1<FN w(tj) THEN  LET m
ay=FN w(tj)
6520 IF a2<=0 THEN  IF FN w(tj)<
=FN v(tj) THEN  LET men=FN w(tj)
6530 IF a2<=0 THEN  IF FN w(tj)>
FN v(tj) THEN  LET men=FN v(tj)
6540 IF a2>0 THEN  IF vo2<=FN v(
tj) THEN  LET men=vo2
6550 IF a2>0 THEN  IF vo2>FN v(t
j) THEN  LET men=FN v(tj)
6600 GO TO 6100
6997
6998
6999
7000 REM portada
7001
7010 PAPER 1: INK 0: CLS
7011 DIM a$(10,1)
7012 FOR n=1 TO 10: READ a$(n):
NEXT n
7013 FOR h=1 TO 3: BORDER 2+h: C
LS
7014 INK 6: PLOT 16,70: DRAW 225
,0
7015 FOR n=1 TO 10
7020 LET i=INT (8*RND)
7025 IF i=1 OR i=0 THEN  GO TO 7
020
7030 LET p=1+INT (10*RND): IF p<
8 THEN  GO TO 7030
7040 PRINT AT p,3+2*n; INK i; FL
ASH 1;a$(n)
7041 PRINT AT 12,3+2*n; INK 4+h;
"[illegible]"
7045 BEEP .1,10+n*5
7050 NEXT n
7061 IF h=3 THEN  PRINT AT 16,1;
 INK 5;Miguel Gisbert Brianso"
;CHR$ 8; OVER 1;"'"
7070 PAUSE 100
7075 CLS
7080 NEXT H
7090 RETURN
8996
```

```
8997
8998
8999
9000 REM caracteres del usuario
9001
9010 FOR i=0 TO 7: READ s: POKE
USR "N"+i,s: NEXT i
9020 FOR i=0 TO 7: READ s: POKE
USR "C"+i,s: NEXT i
9030 FOR i=0 TO 7: READ s: POKE
USR "D"+i,s: NEXT i
9040 FOR i=0 TO 7: POKE USR "a"+
i,24: NEXT i
9050 RETURN
9051
9052
9053
9054
9055 REM DATAS
9060 DATA 24,36,4,8,60,0,0,0
9070 DATA 120,120,255,255,102,0,
0,0
9080 DATA 30,30,255,255,102,0,0,
0
9100 DATA "C","I","N","E","M","A
","T","I","C","A"
9797
9798
9799
9800 REM subrutina de escala
9810 IF tf<=tin THEN  LET escala
=105/FN r(tf): RETURN
9820 IF tf>tin THEN  IF df>=0 TH
EN  LET escala=105/FN r(tin): RE
TURN
9830 IF df<0 THEN  LET escala=10
5/(din-df): LET in=10-FN r(tf)*e
scala: RETURN
9840
9850
9860
9900 REM subrutina de limpiado d
e pantalla
9901
9910 INK 4: PRINT AT 20,2: INVER
SE 1;"Pulse una letra para segui
r": PAUSE 0: CLS : RETURN
```

respondemos a tus

P En vuestro número 12 (octubre 85) se incluye un programa titulado «Editor de pantallas», que subcribe Edu. Salvador. Me interesó mucho ese programa por mi afición al dibujo y la pintura, y muy entusiasmado, procedí según las instrucciones dadas a introducirlo en el ordenador. Cargué la 1.ª parte sin problemas aparentes, la grabé, y procedí pues a teclear la 2.ª parte, la referente a los 765 códigos, en esa parte del programa las dos últimas líneas dicen textualmente: 60 IF sum= 68169 THEN PRINT "OK": STOP, 70 PRINT "ERROR".

Luego de teclear paciente y cuidadosamente los 765 códigos unas cuantas veces, al llegar al último de ellos, aparecía la sentencia "ERROR", procedí pues a sumar con una calculadora todos los códigos, y bendito sea, para mi asombro, la suma daba un total de 68196 y no 68169 como rezaba en la referida instrucción 60. Creí pues resuelto el problema y supuse que se debía a un error de la impresora. Por enésima vez tecleé los sagrados códigos y ¡ALELUYA!, apareció la esperada sentencia "OK" al final. Sentí una alegria inmensa, pues el problema parecía resuelto.

Con ambas partes del programa grabadas y verificadas, me dispuse a cargarlo completo. Al finalizar este proceso, en la pantalla de mi monitor apareció un menú de opciones, y una enorme sonrisa lleno mi rostro. Desafortunadamente duró muy poco, pues al oprimir la tecla 1 (dibujar), de la pantalla desapareció toda la imagen, del computador todo el programa, y de mi rostro la sonrisa. Con todo me dispuse a cargarlo de nuevo, y al oprimir el número 2 (textos) se fue de nuevo todo al diablo, así lo hice repitiendo el proceso una y otra vez con las otras opciones del menú, obteniendo irremediablemente los mismos funestos resultados. Ya caliente (léase molesto e irritado) como mi computador, desistí frustrado.

Quiero suponer que vosotros verificáis los programas que la revista publica mensualmente, quiero decir, que los ponéis en práctica para ver si funcionan y como lo hace. En ello va comprometido el prestigio y la credibilidad de una revista que llega y es apreciada fuera de su frontera a los miles de usuarios en mi país, Venezuela, y el resto de latinoamérica, y que se sentirán igualmente decepcionados luego de pasar largas horas oprimiendo teclas y con la vista fija en la pantalla, esperando dar rienda suelta a su creatividad una vez finalizado el casi siempre tedioso trabajo de copiar un programa como al que aquí hago referencia.

José M. Arría
Caracas (Venezuela)

R **Efectivamente, un error de impresión hizo que uno de los datos que aparecían en el listado, concretamente el correspondiente a la posición de memoria 36957, saliera cambiado, donde aparece un 190 debía haber un 153. De ahí tu confusión y la de algún otro lector que malgastaron «en balde» su tiempo por culpa nuestra. Os pedimos a todos los afectados que sepáis disculpar este error, que al ser de última hora, no fue detectado por nuestros controles de erratas.**

Además, la forma de carga por medio de INPUTs, dada por el autor del artículo, hacía que cada vez que surgiera el fallo hubiera que te-

CARGADOR C/M EDITOR

```
 10 LET sum=0
 20 FOR n=36955 TO 37719
 30 READ a
 40 POKE n,a
 50 LET sum=sum+a
 60 NEXT n
 70 IF sum<>68169 THEN  PRINT
FLASH 1;" ERROR EN DATAs ": STO

 80 PRINT "CARGA OK"
110 DATA 33,88,163,17,88
120 DATA 236,1,0,8,229
130 DATA 213,237,176,209,235
140 DATA 124,214,229,56,6
150 DATA 6,17,62,10,24
160 DATA 4,6,10,62,17
170 DATA 14,0,9,235,225
180 DATA 71,9,6,1,237
190 DATA 176,201,0,0,0
200 DATA 33,22,89,124,230
210 DATA 3,203,39,203,39
220 DATA 203,39,198,64,103
230 DATA 1,88,164,9,6
240 DATA 8,126,229,33,236
250 DATA 89,79,125,197,5
260 DATA 214,32,16,252,111
270 DATA 121,6,8,203,127
280 DATA 40,2,54,0,35
290 DATA 203,39,16,245,193
300 DATA 225,36,16,223,201
310 DATA 42,137,144,124,230
320 DATA 3,203,39,203,39
330 DATA 203,39,198,64,103
340 DATA 1,88,164,9,6
350 DATA 8,229,33,236,89
360 DATA 125,197,5,214,32
370 DATA 16,252,111,14,0
380 DATA 6,8,126,203,33
390 DATA 167,32,2,203,193
400 DATA 35,16,245,121,193
410 DATA 225,119,36,16,222
420 DATA 201,14,33,88,255
430 DATA 58,247,144,203,39
440 DATA 203,39,203,39,22
450 DATA 0,95,25,6,8
460 DATA 126,229,33,236,89
470 DATA 79,125,197,5,214
480 DATA 32,16,252,111,121
490 DATA 6,8,203,127,40
500 DATA 2,54,0,35,203
510 DATA 39,16,245,193,225
```

```
520 DATA 35,16,223,201,33
530 DATA 88,255,58,247,144
540 DATA 203,39,203,39,203
550 DATA 39,22,0,95,25
560 DATA 6,8,229,33,236
570 DATA 89,125,197,5,214
580 DATA 32,16,252,111,14
590 DATA 0,6,8,126,203
600 DATA 33,167,32,2,203
610 DATA 193,35,16,245,121
620 DATA 193,225,119,35,16
630 DATA 222,201,6,8,33
640 DATA 236,89,125,197,5
650 DATA 214,32,16,252,111
660 DATA 1,0,8,203,33
670 DATA 126,35,167,32,2
680 DATA 203,193,16,245,6
690 DATA 8,125,144,111,203
700 DATA 65,40,4,54,0
710 DATA 24,2,54,56,35
720 DATA 203,41,16,241,193
730 DATA 16,208,201,153,90
740 DATA 60,24,60,90,90
750 DATA 60,17,148,145,6
760 DATA 8,33,12,89,125
770 DATA 197,214,32,16,252
780 DATA 111,1,0,8,203
790 DATA 33,126,35,167,32
800 DATA 2,203,193,16,245
810 DATA 121,18,193,19,16
820 DATA 226,17,148,145,6
830 DATA 8,33,20,89,197
840 DATA 45,16,253,26,19
850 DATA 79,203,65,40,4
860 DATA 54,0,24,2,54
870 DATA 56,203,41,125,198
880 DATA 32,111,48,238,193
890 DATA 16,225,201,33,0
900 DATA 64,78,124,6,5
910 DATA 203,39,16,252,167
920 DATA 121,40,4,238,128
930 DATA 24,2,238,255,119
940 DATA 35,124,254,88,200
950 DATA 24,230,0,0,0
960 DATA 33,88,171,17,89
970 DATA 171,54,56,1,255
980 DATA 2,229,237,176,225
990 DATA 1,0,27,9,216
1000 DATA 84,93,19,24,237
1010 DATA 0,0,0,33,15
1020 DATA 91,126,35,34,0
1030 DATA 91,111,60,200,38
1040 DATA 0,41,41,41,237
1050 DATA 75,54,92,9,62
1060 DATA 8,50,4,91,58
1070 DATA 11,91,50,9,91
1080 DATA 58,10,91,50,8
1090 DATA 91,62,9,50,5
1100 DATA 91,126,35,34,2
1110 DATA 91,7,50,6,91
1120 DATA 58,5,91,61,32
1130 DATA 50,58,4,91,61
1140 DATA 32,24,58,14,91
1150 DATA 71,58,12,91,79
1160 DATA 58,10,91,129,5
1170 DATA 32,252,50,10,91
1180 DATA 42,0,91,195,35
1190 DATA 146,50,4,91,58
1200 DATA 13,91,71,58,9
1210 DATA 91,128,50,9,91
1220 DATA 42,2,91,195,64
1230 DATA 146,50,5,91,58
1240 DATA 12,91,71,58,9
1250 DATA 91,50,7,91,58
1260 DATA 13,91,79,197,205
1270 DATA 196,146,193,58,7
1280 DATA 91,60,50,7,91
1290 DATA 13,32,241,58,8
1300 DATA 91,60,50,8,91
1310 DATA 5,32,221,58,6
1320 DATA 91,195,80,146,128
1330 DATA 64,32,16,8,4
1340 DATA 2,1,58,142,92
1350 DATA 238,255,71,58,141
1360 DATA 92,160,71,58,8
1370 DATA 91,230,248,111,58
1380 DATA 7,91,254,192,208
1390 DATA 31,31,31,230,31
1400 DATA 103,203,28,203,29
1410 DATA 203,28,203,29,203
1420 DATA 28,203,29,62,88
1430 DATA 180,103,58,142,92
1440 DATA 166,176,119,58,7
1450 DATA 91,71,230,7,246
1460 DATA 64,103,120,31,31
1470 DATA 31,230,24,180,103
1480 DATA 120,23,23,230,224
1490 DATA 111,58,8,91,71
1500 DATA 31,31,31,230,31
1510 DATA 181,111,235,33,188
1520 DATA 146,120,230,7,79
1530 DATA 6,0,9,70,26
1540 DATA 33,6,91,203,70
1550 DATA 40,3,176,18,201
1560 DATA 47,176,47,18,201
1570 DATA 0,0,0,0,0
1580 DATA 0,0,0,0,0
1590 DATA 0,0,0,0,23
1600 DATA 220,10,206,11,231
1610 DATA 80,26,23,33,88
1620 DATA 228,17,0,64,1
1630 DATA 0,27,237,176,201
```

clear de nuevo todo el código. Intentando subsanar este error incluimos un programa cargador en BASIC que, al incluir los códigos en líneas DATA, hará más fácil la rectificación de los posibles errores que incluyáis.

P Poseo un QL y un Spectrum, en las instrucciones del QL indican que puede establecerse una red entre QL y Spectrum (en este último tengo el Interface 1), pero aunque logro que se transmitan, no se entienden, quizás por el distinto modo de procesar los datos y por sus distintos microprocesadores. La pregunta es la siguiente: ¿Cómo podría establecer una comunicación inteligible entre ambos ordenadores?. Y otra: ¿Hay posibilidad de detener y volver a reiniciar un listado al poner solo LIST? Otros ordenadores lo hacen con CTRL S.

Cristobal C. Mejeras
Santiago de Compostela

R **Efectivamente, parece que hay problemas a la hora de conseguir establecer una red entre Spectrum y QL, los intentos que hemos hecho por conseguirlo han acabado como los tuyos. Sin embargo, hay un remedio de comunicarlos que no da problemas, y es por medio de el RS232. Estamos preparando algo en ese sentido, por lo que te recomendamos que sigas atento a los próximos números de ésta tu revista.**

Si hay un método de detener cualquier salida por pantalla en el QL, se trata de pulsar CTRL y a la vez F5. Haciendo esto se detendrá el listado o cualquier otra cosa hasta que pulsemos alguna tecla.



respondemos a tus

P ¿Cómo puedo en el Spectrum hacer, por ejemplo, el catálogo de un cartucho desde código máquina? He buscado en el listado de la ROM y no hay ninguna rutina que gestione los microdrives.

Raul Morata
Mérida (Badajoz)

R **Nunca podrás encontrar la rutina que buscas si lo haces en la ROM «antigua» del Spectrum, ya que no es en ésta en donde están sino en una «nueva» ROM de 8 K que incorpora el Interface 1. Es en ella donde puedes encontrar todas las rutinas que manejan los microdrives, el RS232, la red local, etc. Si estás interesado en acceder a su listado y desensamblarla, puedes hacerlo pasándola a microdrive con SAVE «NROM» CODE 0,8192 y cargándola posteriormente en la parte alta de la memoria. Como en el momento en que se ejecuta SAVE la nueva ROM está paginada, será ésta la que se grave y no la antigua.**

Para utilizar las rutinas de las ROM del Interface 1 debes usar RST 8 seguido del código de enlace 50 definido con DEFB. La variable (nueva) del sistema situada en las direcciones 23789-90 debe contener la dirección de la subrutina de la nueva ROM que queremos sea llamada. En un CAT debe ser 7256, en la dirección 23766 debe ir el número del drive y en 23768 el número del canal donde queramos que se imprima (2 para la pantalla principal). Si piensas volver al BASIC es conveniente que salves en el stack el par de registros HL.

P Estimados amigos de QL Magazine:

Estoy de acuerdo con vuestro análisis del superBasic, sobre todo en lo de la velocidad, esto es debido a tres limitaciones:

1) La más grave es la reubicación dinámica de los datos y programas, y obligar a direccionar todo lo relativo a A6.

Veamos un ejemplo de código de comparar bytes:

a) Normal: Busca
CMPM.B (A0)+m(A1)+ 4 bytes
DBcc Dn,Busca 4 bytes

b) Relativo a A6: Busca
MOVE.B 0(A6,A0.L),D1 4 bytes
CMP.B D1,0(A6,A1.L) 4 bytes
ADDQ.L 1,A0 2 bytes
ADDQ.L 1,A1 2 bytes
DBcc Dn, Busca 4 bytes

Si se cuenta bytes y se comparan tiempos de ejecución se queda uno espantado, resulta asombrosa la potencia bruta del procesador, para sólo correr de promedio como el MBASIC sobre un 8086 a 8 MHz.

2) Los gráficos son todos en coma flotante, afortunadamente es la «especialidad» del 68008, pero hace perder tiempo.

3) Los programas se salvan sin tokenizar, ya hemos visto lo que hay que hacer para comparar, con lo que el proceso de carga es odioso.

Naturalmente hay soluciones: la mejor es el Simon's Goodwin superBasic Compiler, con el que el aumento promedio de velocidad es de 20 a 100 (en los compiladores de otros ordenadores el aumento es sólo de 5 a 10). Respecto a las cargas hay unas rutinas del grupo de Cuanta, desarrolladas por un suizo, que tragan un programa gigantes en 20 a 30 seg. (lo salvan y cargan tokenizado).

José M. Guzmán
Sevilla

R **Agradecemos su ampliación al artículo «Sobre el superBasic» del pasado mes de noviembre. Aunque se pueda considerar lento al QL si tenemos encuenta el microprocesador que lleva; no hay duda que la mayoría de sus usuarios, acostumbrados a ordenadores domésticos no pensarán así. Seguramente éstos alegaran que será lento para llevar un 68008, pero enormemente rápido para lo que cuesta.**



DIRECTOR:
Enrique F. Larreta
REDACTOR JEFE:
Emiliano Juárez
REDACCION:
Ignacio Borrell, Octavio López, Antonio del Río
DISEÑO:
Ricardo Segura y Benito Gil

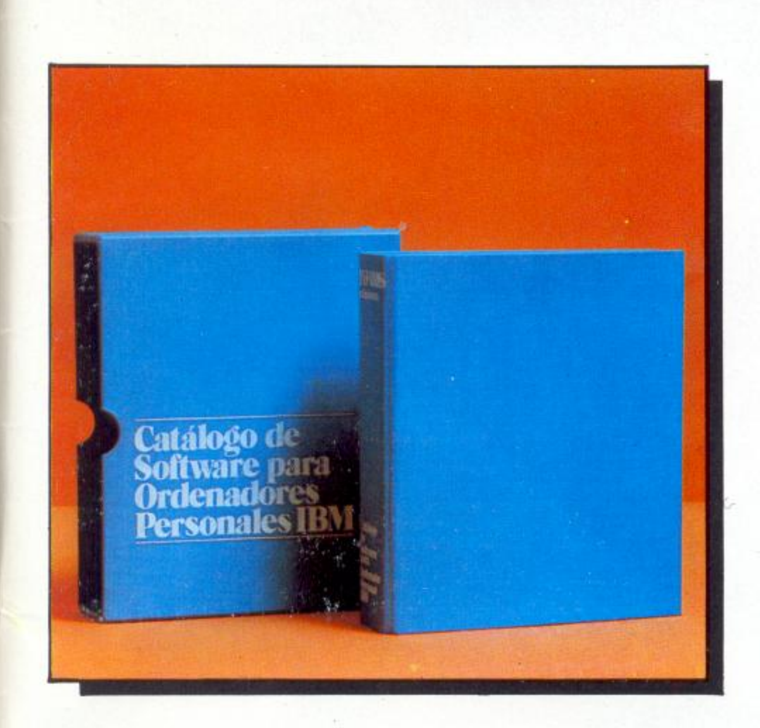
Editado por PUBLINFORMATICA, S. A.
Presidente:
Fernando Bolín
Director Editorial Revistas de Usuarios:
Juan Arencibia
Gerente de circulación y ventas:
Luis Carrero
Producción: Miguel Onieva
Director de Marketing:
Antonio González
Servicio al cliente:
Julia González. Tel.: 733 79 69
Administración:
PUBLINFORMATICA, S. A.
Publicidad:
Emilio García
Dirección y Redacción:
Bravo Murillo, 377. 5.º A. Tel. 733 74 13.
Télex: 48877 OPZX e 28020 Madrid
Administración y Publicidad:
Bravo Murillo, 377. 3.º E. Tels. 733 96 62 - 96
Publicidad Barcelona:
María del Carmen Ríos. Pelayo, 12. Tel. (93) 318 02 89. 08001 Barcelona
Depósito legal: M-29041-1984
Distribuye S.G.E.L. Avda. Valdelaparra, s/n. Alcobendas (Madrid).
Fotomecánica: Karmat, C/ Pantoja, 10. Madrid.
Fotocomposición: Artecomp.
Imprime: Héroes, C/ Torrelara, 8. Madrid.
Distribuidor en VENEZUELA,
SIPAM, S. A.
AVD. REPUBLICA DOMINICANA, EDIF. FELTREC - OFICINA 4B BOLEITA SUR CARACAS (VENEZUELA).
Esta publicación es miembro de la Asociación de Revistas de Información ARI asociada a la Federación Internacional de Prensa Periódica, FIPP.

SUSCRIPCIONES:
Rogamos dirijan toda la correspondencia relacionada con suscripciones a:
TODOSPECTRUM EDISA: Tel. 415 97 12
C/ López de Hoyos, 141-5º
28002 MADRID
(Para todos los pagos reseñar solamentes TODOSPECTRUM)
Para la compra de ejemplares atrasados diríjanse a la propia editorial
TODOSPECTRUM
C/ Bravo Murillo, 377. 5.º A
Tel. 733 74 13 - 28020 MADRID

Si deseas colaborar en TODOSPECTRUM remite tus artículos o programas a Bravo Murillo, 377. 5.º A. 28020 Madrid. Los programas deberán estar grabados en cassette y los artículos mecanografiados. A efectos de remuneración, se analiza cada colaboración aisladamente, estudiando su complejidad y calidad.

CATALOGO DE SOFTWARE PARA ORDENADORES PERSONALES IBM

TODO EL CATALOGO DE SOFTWARE CON MAS DE 800 FICHAS





1.ª ENTREGA 3.500,— PTAS.
(400 FICHAS + FICHERO)

RESTO EN TRES ENTREGAS TRIMESTRALES DE 1.500,— PTAS. CADA UNA.

PRECIO TOTAL DE LA SUSCRIPCION - 8.000,— PTAS.

CUPON DE PEDIDO

SOLICITE **HOY MISMO** EL CATALOGO DIRECTAMENTE A

infodis, s.a.

BRAVO MURILLO, 377 - 5.° A
28020 MADRID

O EN LOS CONCESIONARIOS IBM

El importe lo abonaré: POR CHEQUE ☐ CONTRA REEMBOLSO ☐
CON MI TARJETA DE CREDITO ☐ Ref: CATALOGO DE SOFTWARE

Cargue **8.000** ptas. a mi tarjeta American Express ☐ Visa ☐ Interbank

Número de mi tarjeta ______

Fecha de caducidad ______ Firma

NOMBRE ______

CALLE ______

CIUDAD ______ D.P. ______

PROVINCIA ______